1998-06

AUV fault detection using model based observer residuals

Melvin, James E.

Monterey, California. Naval Postgraduate School

NPS-ME-98-004

NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFORNIA

THESIS

AUV FAULT DETECTION USING MODEL BASED OBSERVER RESIDUALS

by

James E. Melvin

June 1998

Thesis Advisor: Anthony J. Healey

Approved for public release; distribution is unlimited.

Prepared for:
Office of Naval Research,
Arlington, Virginia 2217-5660

NAVAL POSTGRADUATE SCHOOL
Monterey, California 93943-5000

Rear Admiral Robert C. Chaplin
Superintendent

This thesis was prepared in conjunction with research sponsored in part by the Office of Naval Research, Arlington, Virginia, under N0001498WR20016.

Released by:

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE June 1998	3. REPORT TYPE AND DATES COVERED Engineer's and Master's Thesis

4. TITLE AND SUBTITLE : AUV FAULT DETECTION USING MODEL BASED OBSERVER RESIDUALS

5. FUNDING NUMBERS N0001498WR20016

6. AUTHOR(S) JAMES E. MELVIN

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER NPS-ME-98-004

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Office of Naval Research, 800 N. Quincy St., Arlington, VA 22217-5660

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE:

13. ABSTRACT *(maximum 200 words)*

In order for the Navy's next generation Unmanned Undersea Vehicles to be more robust to software/hardware faults, on-line failure detection and resolution is needed. Typically, fault detection methods include limits and trends analysis, model free, and model based techniques. Here, model based observers are proposed for the detection of fault induced dynamic signals in the diving, steering, and roll control systems. Such automatic fault detection systems were designed and implemented in a *Simulink* model of the "21UUV." In the course of conducting simulations with the model, numerous vehicle behaviors were studied and detection response was verified. In addition, the model based observer residuals may be designed to distinguish actuator faults from wave disturbances and fin faults from maneuvering responses.

14. SUBJECT TERMS: MODEL BASED OBSERVERS, 21UUV, AUVs, FAULT DETECTION

15. NUMBER OF PAGES 134

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18 298-102

AUV FAULT DETECTION USING MODEL BASED OBSERVER RESIDUALS

James E. Melvin
Lieutenant, United States Navy
B.S., United States Naval Academy, 1989
M.B.A., National University, 1995

Submitted in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING
and
MECHANICAL ENGINEER

from the

NAVAL POSTGRADUATE SCHOOL
June 1998

ABSTRACT

In order for the Navy's next generation Unmanned Undersea Vehicles to be more robust to software/hardware faults, on-line failure detection and resolution is needed. Typically, fault detection methods include limits and trends analysis, model free, and model based techniques. Here, model based observers are proposed for the detection of fault induced dynamic signals in the diving, steering, and roll control systems. Such automatic fault detection systems were designed and implemented in a *Simulink* model of the "21UUV." In the course of conducting simulations with the model, numerous vehicle behaviors were studied and detection response was verified. In addition, the model based observer residuals may be designed to distinguish actuator faults from wave disturbances and fin faults from maneuvering responses.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS

I would like to acknowledge the Office of Naval Research for providing financial support for this research. Additionally, I would like to express my sincere appreciation to Professor Tony Healey for his guidance, patience and motivation throughout the thesis process. His high level of technical competence and uncanny ability to teach has provided me with the resources necessary to solve any type of problem in the future. Finally and most importantly, I would like to thank my wife, Karen, and son, Justin, for their love and support.

I. INTRODUCTION

A. GENERAL BACKGROUND AND LITERATURE

The increased desire to use Autonomous Underwater Vehicles (AUVs) and Unmanned Undersea Vehicles (UUVs) for commercial and military applications has led to a great deal of research in this field over the last decade. The military, as well as industry, can see the great potential uses for AUVs and UUVs in the oceanic environment. Although great strides have been made in this field of research, to develop both methodology and technology pieces, further work is needed in precision navigation, sensor development and integration, and especially improving the reliability of long term mission completion.

Previous work on AUV technology shows that underwater navigation to sufficient precision within cost limits is possible. AUV uses for oceanographic survey has been described in Bellingham (1997) have given results on positioning accuracy for survey missions. Marco and Healey (1996) have demonstrated a method to navigate an AUV in a local area using an acoustic sensor for position information derived from feature detection. Healey and Lienard (1993) have shown that multivariable sliding mode autopilot based on state feedback, designed assuming decoupled modeling, is quite satisfactory for the combined diving, steering, and speed response of a slow speed AUV. Cristi, Healey, and Papoulias (1990) have illustrated that adaptive sliding mode control in the dive plane is possible for AUVs. Now that cost effective, precise underwater navigation is becoming possible, the remaining major technical issue is the overall reliability of the vehicle for long term and complex missions.

In order for AUVs and UUVs to be a more robust, self-sufficient system, an on-line failure detection and resolution system is needed. The failure detection and resolution system must work in tandem with an AUV or UUV whose systems are reconfigurable, so that minor faults may be overcome by control reconfiguration. This area of study, commonly linked to failure modes and effects analysis (FMEA), has received a great deal of attention the past several years. Previous literature pertinent to this research is abundant since the aircraft, spacecraft, and process industries have all written about fault detection techniques. A brief summary of some basic fault detection techniques along with some examples can be found in Gertler (1986). Bell, et al., (1992) has developed a tool that automates the reasoning portion of an FMEA. The prototype has been created and successfully passed a test and evaluation program. Healey (1992) has addressed the use of Kalman Filters and Artificial Neural Networks to provide detection, and isolation of impending subsystem failures. Finally, Hurni (1997) shows that *Simulink* can be used as a modeling and simulation tool for FMEA on an AUV steering system. In real time control of autonomous systems, however, failure detection and accommodation is required as part of an overall system controller.

This study concentrates in a specific area of fault detection analysis; the use of model based observers for fault detection. Previous literature in this particular area is also abundant.

Early approaches to analytical redundancy for fault detection have been described in the surveys by Wilsky (1976) and Isermann (1984), more recently, Patton (1997). The system model includes models of the anticipated faults - often as an additive input or output, or as a multiplicative factor as in a parameter change. Use of a bank of Kalman

Filters, each "tuned" to a particular fault are then used to generate a "residual" – the innovation (see Napolitano and Swain (1989) for an aircraft application). The statistical properties of the residual are then tested against either single or multiple hypotheses and the residual with the maximum likelihood compared to a threshold is selected as the identifier of a fault. If no faults occur, all residuals are zero.

Problems with the above approaches are that the model based filter residuals are sensitive to both faults and disturbances. In fact, they are also sensitive to unmodeled dynamics and coupling inputs from other response modes in the system. Also, while the observer is less sensitive to input commands than a servo error detector, there are maneuvering responses that occur even when faults are absent.

Eigenstructure analysis, Speyer (1987) and Patton and Chen (1991), solves the problem of eigenvector as well as eigenvalue design in the residual generation system such that isolatable disturbance responses can be distinguished from fault responses, and a weighted residual measure can lead to either left or right eigenvector consideration. Alternatively, the fault model is embedded into the filter and the fault state is identified as a state output from the filter - its mean, together with its covariance, so that again likelihood measures can be assessed. This approach is described in Mangoubi et. al., (1995) with specific application to an underwater vehicle, who proposed a "robust" likelihood measure for separating unmodelled dynamics from an actuator fault.

Still, the problem of robustness especially to intermode coupling, simplicity of software management, and threshold design, is problematic when it is considered that future UUV missions will require long term reliability for shallow water operations in energetic (high sea states) environments. Healey (1998) has illustrated a proposed fault

detection architecture that weights inputs from servo error detectors, wave action detectors, residual generators, into a fuzzy inference systems that is conjectured to provide robustness and simplicity through decomposition. This thesis and related work is a first step, aimed at demonstrating the correctness of that assertion.

B. SCOPE OF THIS WORK

The overall problem of autonomous fault detection is complex and diverse. This study will focus on sensor based fault detection limited to the primary subsystems of the Naval Undersea Warfare Center's experimental UUV (21UUV), including the diving, steering, roll and speed control systems. In particular, model based observer residuals for the detection of fin faults and weight buoyancy mismatches in shallow water operations with wave effects is discussed. The purpose of this thesis is threefold:

1. To design model based observers for the diving, steering, and roll control systems of the 21UUV;

2. To implement the model based observers into a computer model of the 21UUV; and

3. To run simulations on the 21 UUV computer model to determine if model based observers can robustly detect fin faults and weight-buoyancy mismatches in a shallow water environment.

Chapter II discusses different types of faults that can occur on UUVs. Most faults can be distinguished as hardware/software problems on the vehicle, although some environmental disturbances can be defined as a "fault" since they jeopardize mission completion. In addition, advantages and disadvantages of different types of fault detection techniques are described including limits and trends analysis, model free detection, and model based detection.

Chapter III describes a proposed fault detection architecture of 21UUV and discusses the design of model based observers for the diving, steering and roll control systems. The design of each model based observers is divided into a "theory" and a "application" section. The application section shows how each observer was implemented into the 21UUV computer model.

Chapter IV is intended to be a "User's Guide" for the *Simulink* 21UUV computer model. A brief description of each section is given and appropriate *Matlab* files are referenced. A detailed description on how to input data into the program for successful simulations is also included.

Chapter V shows results that prove model based observer residuals can be used for the detection of fin faults and weight-buoyancy mismatches in the shallow water environment. Simulations were conducted on the 21UUV model to verify that fin faults could be distinguished from maneuvering responses, and that fin faults could be distinguished from wave disturbances.

Chapter VI lists the conclusions of this report derived for the results of Chapter V. In addition, recommendations are made for additional study.

II. FAULT TYPES AND DETECTION / DIAGNOSTIC TECHNIQUES

Long term deployments of autonomous systems in the ocean require replenishment of energy supplies and reliable fault free operation. It is recognized that fault free operation will not always be possible, so that system design must pay attention to a study of failure modes and effects. The purpose of this chapter is to describe different types of faults and possible detection methods common to the UUV.

A. FAULT TYPES

In spite of good engineering practice, faults can occur. Two kinds of faults can be identified:

1. Those that arise from malfunctions in the hardware and software subsystems in the vehicle; and

2. Those that arise from environmental conditions that are viewed as disturbances, and while these may not be direct malfunctions, they have the effect of performance degradation and the completion of a mission is jeopardized.

Even though UUVs are usually equipped with highly reliable sensor suites, connections and computing components, hardware and software faults can occur in the vehicle. An example of a hardware fault would be the loss of steering resulting from a stuck or loose fin.

Since the UUVs sensors and computing components are normally highly reliable and accurate, environmental conditions are easily detected by the vehicle. These disturbances may cause significant variations in inertial velocities as well as translational and rotational measurements. An example of an environmental condition "fault" would

be the inability of the vehicle to take a data measurement because of a high sea state in shallow water operations.

Another way to classify faults is by time. Incipient faults occur when a signal or system gradually degrades over a long period of time. Incipient faults are difficult to detect because the signal varies very slowly over time.

Abrupt faults are those that usually cause a "jump" variation in the signal in a short period of time. These are the types of signals that can be more easily detected to quickly determine actuator, sensory and plant faults on UUVs. Examples include a sudden loss of power to a sensor, or a sudden loss of a mechanical linkage (propeller breaks free).

B. DETECTION / DIAGONOSTIC TECHNIQUES

Many papers have been written on how to design a system that will automatically detect the presence of a fault. Research is not limited to just UUVs. The aircraft, spacecraft, and process industries have also written about fault detection techniques. As described by Gertler (1986), fault detection methods can be classified into three methods: those that use simple limits and trends analysis, those that use detection techniques which are without the use of analytical models, and those that provide models as the basis for detection filters.

1. Limits and Trends Analysis

A survey of fault detection methods indicates that alarms can be easily monitored if signals are static. This is done using "limits and trends" analysis. In this method, the

measured signal is compared to a previously set threshold. Exceeding the threshold would indicate a fault condition that would be passed to a higher level controller for subsequent action. Limits and trends analysis is suitable for static or slow varying signals such as computer bay temperatures and battery voltage, Fossen (1994).

2. Model Free Detection

Limits and trends analysis is not suitable for the detection of dynamic signals such as wave action. Dynamic signals tend to exceed threshold limits, only to come back within bounds a short time later. This causes thresholding alone to be a problem.

Model free methods attempt to extract a constant feature of a signal to compare against a threshold value. This is the case when a spectral analysis is performed on a signal and spectral levels in specified frequency bands can be compared against thresholds. The model free method is useful to detect the presence of frequency components in servo error signals and could be used to identify levels of wave induced disturbances considered as faults, Newland (1993).

3. Model Based Residual Generation

Model free methods have difficulty detecting dynamic signals developed from autopilot errors. These error signals are naturally large when steering to new commands, but small if correct final heading is maintained. In addition, wave motion causes wave period oscillatory motion in servo errors.

Model based methods have been found to be useful over the last 20 years or so in detecting dynamic signals embedded in noise. By using a model based observer, a

"residual" can be generated between the sensor measured values and that predicted for the model. The residual provides a signal that is not sensitive to servo errors caused by command changes, and responds primarily to non ideal loads, disturbances from waves, and sensor signal errors. Likelihood functions are then used to provide probability ratios corresponding to fault based hypotheses. Simple or multiple hypothesis testing is done with decision making on the basis of threshold exceedance. The List of References gives a listing of papers relevant to the methods, Isermann (1984), Patton (1997), Speyer (1987), Willsky (1976), Beard (1971) and Jones (1973).

III. FAULT DETECTION ARCHITECTURE AND OBSERVER DESIGN

A. PROPOSED FAULT DETECTION ARCHITECTURE FOR 21UUV

The overall problem of autonomous fault detection for 21UUV and similar Navy vehicles is complex. Therefore, the focus of sensor based fault detection will be limited to the primary subsystems of the vehicle, including the diving, steering, speed control, roll control, navigation, powering, and computer subsystems (sensor outputs from the vehicle as opposed to environmental sensors such as forward looking or side scan sonar). As far as motion capabilities are concerned, the diving, steering, speed and roll control subsystems should each be monitored. Faults that impair the capability of these subsystems need to be detected so that reconfiguration of control settings may help to mitigate a premature mission abort. Some graceful degradation of mission performance could be allowed if "partial" depth control or speed control could be maintained.

The proposed fault detection architecture for 21UUV consists of sensor outputs from the vehicle, controllers, fault detectors and a fuzzy inference system. Sensor outputs from the vehicle feed into the controllers and the fault detectors. The controllers provide control inputs back into the vehicle and to the set of fault detectors. The fault detectors compare inputs from the sensor outputs and the controller and use residual generators to determine if a fault has occurred. If a fault has occurred, the signal is passed to the fuzzy inference system to determine the accommodating response. The fuzzy inference system is aimed at providing robustness and determines whether the fault can be handled by making a command adjustment to the controllers or a health assessment to a higher level control is needed. Because residual generation is known to

be sensitive to unmodelled inputs (coupling from other modes of response and disturbances, as well as faults, the weighting of inputs from several observation systems is expected to increase the reliability of fault detection. Figure 3.1 shows how this is illustrated.

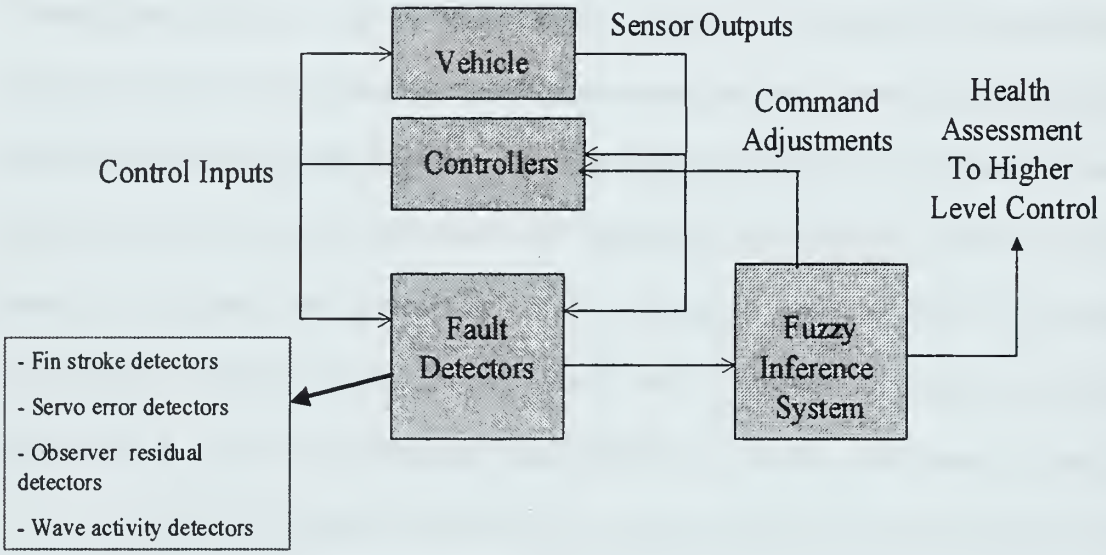

Figure 3.1 Proposed Fault Detection Architecture for 21 UUV (Healey, 1998)

Four different sets of fault detectors are used in this architecture to improve robustness: fin stroke detectors, servo error detectors, observer residual detectors, and wave activity detectors. The fin stroke and servo error detectors use simple detection circuits that look for signal magnitude as well as length of time persisting in the fault mode that produces the fault declaration. These are model free detectors. An observer residual detector is a model-based detector that is used to generate a "residual" between the sensor measured values and that predicted from the model using the same control input as applied to the vehicle. The residual magnitude is processed and compared to a

threshold value to determine exceedance. Then, the residual is processed through a non-linear filter, with a forgetting factor, to determine the persistence of the fault. Finally, the level of severity of the fault is determined through quantization. An example of this exceedance and persistence scheme is shown in Figure 3.2.

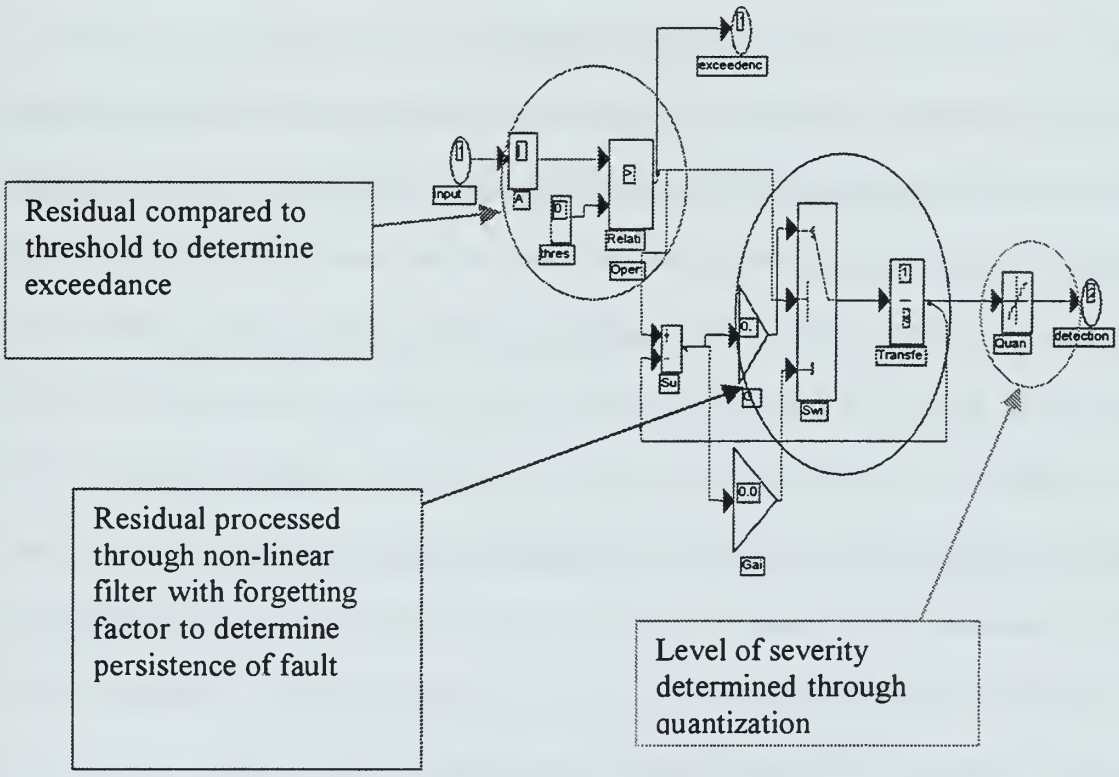

Figure 3.2 Exceedance and Persistence Detection (*Simulink* Model, Healey, 1998)

The fault detection architecture for 21UUV must also take into account operation near the surface under waves. By comparison of residual signals, a distinction can be made between wave disturbance and an actuator fault condition. Robustness is improved when a wave activity detector is added. The combination of all sources of information allows for the accommodation response using a fuzzy inference system linking residual

fault declaration signals as inputs that are mapped to recommended actions using fuzzy rules.

B. DESIGN OF MODEL BASED RESIDUAL OBSERVERS

Dynamic signals such as those developed by autopilot errors become difficult to detect, as errors are naturally large when steering to new commands, yet are small if correct final heading is maintained. Wave motion causes wave period oscillatory motion in servo errors. As described in earlier chapters, many methods are available for fault detection, but model based methods, like observer residual detectors, have been found to be useful in detecting dynamic faults and are the subject of this work. Faults in this context, can arise from a bad sensor as well as a faulty actuator (fin, propeller), and may be modeled as an added force or sensor output - generally of unknown magnitude - the presence of which may be detected by residual analysis. The residual provides a signal that is less sensitive to servo errors caused by command changes, and responds primarily to non ideal loads, disturbances from waves, and sensor signal errors. Analysis of the residuals provides the key to subsystem failure detection. The design of the 21UUV diving, steering, and roll control observer residuals detectors will be discussed, using the method outlined in Healey (1998) and Patton and Chen (1991).

The assumption on which our approach to fault detection is based is that a UUV, with six degrees of freedom, may be controlled by four main subsystems that – in the ideal case – are uncoupled. Coupling is known to exist, but is ignored in the approach and assumed to be negligible for the purposes of control system design.

The assumption is that there are four autopilot controllers – the steering, diving, roll and speed control systems, respectively. Because of that assumption, it follows that there should be four observer based residual generators – one for each controller – that would generate, and process residuals for each subsystem.

Each subsystem is modeled as a non interacting LTI system:

$$S:(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})\in \mathbf{R}^{mxn}$$

where it is allowed that the parameters could be speed dependent and either robust control design methods are used, or the autopilots could be 'gain scheduled.'

1. Diving Observer Residuals Detector

a. *Theory*

The diving subsystem dynamics for the 21UUV are modeled by the following equations:

$$\mathbf{x}'(t) = [\hat{w}_r(t), \hat{q}(t), \hat{\theta}(t), \hat{Z}(t)];\ u(t) = \delta_s(t) \text{ with}$$
$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}u(t) + \mathbf{E}\mathbf{f}_a(t) + \mathbf{F}\mathbf{d}(t);$$
$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{f}_s(t);$$

w_r, q, θ, and Z are the heave velocity of the vehicle relative to the water, the pitch rate, the pitch angle, and the depth. **B** and **E** are input vectors for the control planes and added forces and moments caused by the imbalance of commanded and actual loads on the vehicle caused by actuator faults. **F** is the input vector associated with disturbances from waves and currents. It is important to note that since the relative velocity definition for vehicle states is used, **E** and **F** are distinct (i.e. $\mathbf{E}'\mathbf{F} = 0$) with the

result that disturbances from waves and currents are distinguishable from actuator faults [Patton and Chen (1991)]. With a high quality inertial system, all state variables are measured with little noise. Thus, the output matrix, **C**, may be considered to be identity. $\mathbf{f}_a(t)$ and $\mathbf{f}_s(t)$ are considered to be added forces caused by fin faults, and sensor errors respectively.

From the subsystem dynamics, a model based observer can be formed:

$$\hat{\mathbf{x}}'(t) = [\hat{w}_r(t), \hat{q}(t), \hat{\theta}(t), \hat{Z}(t)];\ u(t) = \delta_s(t)\ \text{with}$$
$$\dot{\hat{\mathbf{x}}}(t) = (\mathbf{A} - \mathbf{KC})\hat{\mathbf{x}}(t) + \mathbf{B}u(t) + \mathbf{K}\mathbf{y}(t);$$
$$v(t) = \mathbf{y}(t) - \mathbf{C}\hat{\mathbf{x}}(t);$$

The residuals, $v(t)$, are the differences between the sensor measured states and the model based estimates from the observer. It follows that, the state observation error, ε_x, is given by:

$$\dot{\varepsilon}_x(t) = (\mathbf{A} - \mathbf{KC})\varepsilon_x(t) + \mathbf{E}\mathbf{f}_a(t) + \mathbf{F}\mathbf{d}(t) + \mathbf{K}\mathbf{f}_s(t);$$
$$v(t) = \mathbf{C}\varepsilon_x(t) + \mathbf{f}_s(t);$$

The residuals, $v(t)$, can be expressed as the sum of effects arising from the natural dynamics of the observation system plus responses to actuator or other additive forces embodied in $\mathbf{f}_a(t)$, disturbances $\mathbf{d}(t)$, and sensor signal faults, $\mathbf{f}_s(t)$.

If $\mathbf{C} = \mathbf{I}$, then a residual is produced for each state and , if disturbances and faults are time invariant, the steady state (no transient) response of state residuals is given by:

$$v(\infty) = \mathbf{A}_o^{-1}\mathbf{E}\mathbf{f}_a(\infty) + \mathbf{A}_o^{-1}\mathbf{F}\mathbf{d}(\infty) + \mathbf{A}_o^{-1}\mathbf{K}\mathbf{f}_s(\infty)$$

Now, ignoring sensor faults in this work, the following is observed:

1. The influence of $\mathbf{f}_a$ in a residual $v_i = \mathbf{C}_i v$ is nulled if $\mathbf{C}_i\mathbf{A}_o^{-1}\mathbf{E} = 0$. $\mathbf{A}_o^{-1}$ is diagonal and $\mathbf{C}_i\mathbf{E} = 0$ (i.e. $\mathbf{C}_i$ lies in null$[(\mathbf{A}_o^{-1}\mathbf{E})']$);

2. The influence of $\mathbf{d}$ is a residual $v_i = \mathbf{C}_i v$ and is nulled if $\mathbf{C}_i\,\mathbf{A}_o^{-1}\mathbf{F} = 0$.

The design key is to select $\mathbf{K}_o$ such that $\mathbf{A}_o$ has real eigenvectors. $\mathbf{A}_o$ is diagonal and $\mathbf{C}_i$ is chosen to lie in the null space of $[\mathbf{A}_o^{-1}\mathbf{K}']'$ to be orthogonal to $\mathbf{E}$ to suppress disturbances and to $\mathbf{F}$ for actuator faults, and to $\mathbf{K}$ to suppress sensor faults. Unlike state observation, residual generation requires a different balance in the choice of filter gains, $\mathbf{K}_o$. Too high of a choice leads to the residual being small, dominated by sensor faults and noise. A lower gain set is needed consistent with bandwidth requirements. Of course, stability must be obtained.

The residuals can be analyzed in the frequency domain by:

$$v(s) = \mathbf{C}(s\mathbf{I} - \mathbf{A}_c)^{-1}\mathbf{E}f_a(s) + \mathbf{C}(s\mathbf{I} - \mathbf{A}_c)^{-1}\mathbf{F}d(s) + \mathbf{C}(s\mathbf{I} - \mathbf{A}_c)^{-1}\mathbf{K}f_s(s);$$
$$\text{where} \quad \mathbf{A}_c = (\mathbf{A} - \mathbf{KC})$$

One observer design (low gain) should therefore find a gain set that maximizes the influence of actuator faults, minimizes the influence of wave disturbance, and the sensor faults.

b. ***Application***

Using the 21UUV model (described in chapter four) running at a forward speed of 6 ft/sec, $\mathbf{x}' = [w, q, \theta, Z]$. For a slow forward speed of 6 feet per second, the dynamics and input matrices are:

$$\mathbf{A} = \begin{bmatrix} -0.1140 & 2.3282 & -0.0019 & 0 \\ 0.0649 & -0.3015 & 0.0109 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & -6 & 0 \end{bmatrix} ; \quad \mathbf{B} = \begin{bmatrix} -0.3308 \\ -0.1224 \\ 0 \\ 0 \end{bmatrix}$$

In this example, the vehicle is moving forward with a command to dive. Wave amplitude is set at 2 ft. The autopilot for depth control is a sliding mode design. The placed poles are selected to include a single pole at the origin (required by method), yielding:

$$\lambda = \begin{bmatrix} -0.4 & -0.41 & -0.42 & 0 \end{bmatrix}$$
$$k = \begin{bmatrix} -0.5100 & -5.2770 & -2.0321 & 0 \end{bmatrix}$$
$$s' = \begin{bmatrix} 0.0663 & -0.7046 & -0.7062 & 0.0205 \end{bmatrix}$$

and the control law (Healey, Lienard, 93):

$$\delta_s(t) = -kx + \eta \; satsgn\left(\frac{\sigma}{\varphi}\right)$$

Several different options can be used for the selection of the observer gain, $\mathbf{K_o}$. One option is to use a linear quadratic estimator. This method leads to undesired complex poles and eigenvectors. A better solution is to use real pole placement. This guarantees real poles and eigenvectors. The observer gains are found using the *Matlab* 'place' algorithm to put the observer poles at real values close to the control poles, [-0.2, -0.21, -0.22, -0.24]. The solution for $\mathbf{K_o}$ yields:

$$\mathbf{K}_o = \begin{bmatrix} 0.0860 & 2.3282 & -0.0019 & 0 \\ 0.0649 & -0.0915 & 0.0109 & 0 \\ 0 & 1 & 0.2200 & 0 \\ 1 & 0 & -6 & 0.2400 \end{bmatrix}$$

The observer is modeled by the state-space equation:

$$x'(t) = \mathbf{A}_o x(t) + \mathbf{B}_o u(t)$$
$$y(t) = \mathbf{C}_o x(t) + \mathbf{D}_o u(t)$$
where :
$$\mathbf{A}_o = \mathbf{A} - \mathbf{K}_o' * \mathbf{C}$$
$$\mathbf{B}_o = [\mathbf{B} \quad \mathbf{K}_o']$$
$$\mathbf{C}_o = -\mathbf{C}$$
$$\mathbf{D}_o = [\mathbf{zeros(4,1)} \quad \mathbf{eye(4,4)}]$$

For the diving observer residuals detector:

$$\mathbf{A}_o = \begin{bmatrix} -0.20 & 0 & 0 & 0 \\ 0 & -0.21 & 0 & 0 \\ 0 & 0 & -0.22 & 0 \\ 0 & 0 & 0 & -0.24 \end{bmatrix}$$

$$\mathbf{B}_o = \begin{bmatrix} -0.3308 & 0.0860 & 2.3282 & -0.0019 & 0 \\ -0.1224 & 0.0649 & -0.0915 & 0.0109 & 0 \\ 0 & 0 & 1 & 0.2200 & 0 \\ 0 & 1 & 0 & -6 & 0.2400 \end{bmatrix}$$

$$\mathbf{C}_o = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \quad ; \quad \mathbf{D}_o = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The *Matlab* file used to design the observer, "dive_obs_des", is included in Appendix A.

2. Steering Observer Residuals Detector

a. *Theory*

The steering subsystem dynamics for the 21UUV may be modeled by the following equations:

$$\mathbf{x}'(t) = [\hat{v}_r(t), \hat{r}(t), \hat{\psi}(t)];\ u(t) = \delta_s(t)\ \textit{with}$$
$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}u(t) + \mathbf{E}\mathbf{f}_a(t) + \mathbf{F}\mathbf{d}(t);$$
$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{f}_s(t);$$

v_r, r, and ψ are the sway velocity of the vehicle relative to the water, the yaw rate, and the yaw angle. Besides the different variables listed above, the steering observer is designed in the same matter as the diving observer.

b. *Application*

Using the 21UUV model (described in chapter four) running at a forward speed of 6 ft/sec, $\mathbf{x}'=[v_r, r, \text{and } \psi]$. For a slow forward speed of 6 feet per second, the dynamics and input matrices are:

$$\mathbf{A} = \begin{bmatrix} -0.1140 & -2.3282 & 0 \\ -0.0649 & -0.3015 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad ; \quad \mathbf{B} = \begin{bmatrix} 0.3308 \\ -0.1224 \\ 0 \end{bmatrix}$$

In this example, the vehicle is moving forward with a command to steer. Wave amplitude is set at 2 ft. The autopilot for depth control is a sliding mode design.

The placed poles are selected to include a single pole at the origin (required by method), yielding:

$$\lambda = [-0.4 \quad -0.41 \quad 0]$$
$$k = [0.5762 \quad -1.6663 \quad 0]$$
$$s' = [0.0164 \quad 0.8804 \quad 0.4740]$$

The observer gains are found using the *Matlab* 'place' algorithm to put the observer poles at real values close to the control poles, [-0.2, -0.21, -0.22]. The solution for $\mathbf{K_o}$ yields:

$$\mathbf{K_o} = \begin{bmatrix} 0.0860 & -2.3282 & 0 \\ -0.0649 & -0.0915 & 0 \\ 0 & 1 & 0.2200 \end{bmatrix}$$

The observer is modeled by the state-space equation:

$$x'(t) = \mathbf{A_o} x(t) + \mathbf{B_o} u(t)$$
$$y(t) = \mathbf{C_o} x(t) + \mathbf{D_o} u(t)$$
where :
$$\mathbf{A_o} = \mathbf{A} - \mathbf{K_o}' * \mathbf{C}$$
$$\mathbf{B_o} = \begin{bmatrix} \mathbf{B} & \mathbf{K_o}' \end{bmatrix}$$
$$\mathbf{C_o} = -\mathbf{C}$$
$$\mathbf{D_o} = [\mathbf{zeros}(3,1) \quad \mathbf{eye}(3,3)]$$

For the steering observer residuals detector:

$$\mathbf{A}_o = \begin{bmatrix} -0.2000 & 0 & 0 \\ 0 & -0.2100 & 0 \\ 0 & 0 & -0.2200 \end{bmatrix}$$

$$\mathbf{B}_o = \begin{bmatrix} 0.3308 & 0.0860 & -2.3282 & 0 \\ -0.1224 & -0.0649 & -0.0915 & 0 \\ 0 & 0 & 1 & 0.2200 \end{bmatrix}$$

$$\mathbf{C}_o = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \; ; \quad \mathbf{D}_o = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The *Matlab* file used to design the observer, 'steer_obs_des', is included in Appendix A.

3. Roll Observer Residuals Detector

a. *Theory*

The roll subsystem dynamics for the 21UUV may be modeled by the following equations:

$$\mathbf{x}'(t) = [\hat{\varphi}(t), \hat{p}(t)]; \; u(t) = \delta_s(t) \; with$$
$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}u(t) + \mathbf{E}\mathbf{f}_a(t) + \mathbf{F}\mathbf{d}(t);$$
$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{f}_s(t);$$

ϕ and p are the roll angle and roll rate. Besides the different variables listed above, the roll observer is designed in the same matter as the diving observer.

b. ***Application***

Using the 21UUV model (described in chapter four) running at a forward speed of 6 ft/sec, $\mathbf{x}' = [\phi \text{ and } p]$. For a slow forward speed of 6 feet per second, the dynamics and input matrices are:

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -1.7474 & 0 \end{bmatrix} \; ; \; \mathbf{B} = \begin{bmatrix} 0 \\ 1.4197 \end{bmatrix}$$

In this example, the vehicle is moving forward with a command to steer. Wave amplitude is set at 2 ft. The autopilot for depth control is a sliding mode design. The placed poles are selected to include a single pole at the origin (required by method), yielding:

$$\lambda = \begin{bmatrix} -1.4 & -1.41 \end{bmatrix}$$
$$k = \begin{bmatrix} -1.2308 & 0.9861 \end{bmatrix}$$
$$s' = \begin{bmatrix} 0.8137 & 0.5812 \end{bmatrix}$$

The observer gains are found using the *Matlab* "place" algorithm to put the observer poles at real values close to the control poles, [-1.4, -1.41]. The solution for $\mathbf{K_o}$ yields:

$$\mathbf{K_o} = \begin{bmatrix} 1.4000 & -1.7474 \\ 1 & 1.4100 \end{bmatrix}$$

The roll observer is sensitive to centrifugal force action during steering maneuvers. Thus, a feed forward control system, G_{ff}, was added to $\mathbf{B_o}$:

$$G_{ff} = \frac{mk_f U}{I_x};$$

where :

$m = 88.95 \text{ slugs}$

$k_f = 0.02$

$U = 6 \text{ ft/sec}$

$I_x = 32.78$

m is the mass of the vehicle, k_f is a gain constant, U is the design velocity of the observer, and I_x is a dimensionless coefficient. The observer is modeled by the state-space equation:

$$x'(t) = \mathbf{A}_o x(t) + \mathbf{B}_o u(t)$$
$$y(t) = \mathbf{C}_o x(t) + \mathbf{D}_o u(t)$$

where :

$$\mathbf{A}_o = \mathbf{A} - \mathbf{K}_o' * \mathbf{C}$$
$$\mathbf{B}_o = \left[\mathbf{B} \quad \mathbf{K}_o' \quad (0, G_{ff})'\right]$$
$$\mathbf{C}_o = -\mathbf{C}$$
$$\mathbf{D}_o = \left[\mathbf{zeros}(2,1) \quad \mathbf{eye}(2,2) \quad \mathbf{zeros}(2,1)\right]$$

For the roll observer residuals detector:

$$\mathbf{A}_o = \begin{bmatrix} -1.40 & 0 \\ 0 & -1.41 \end{bmatrix} \; ; \; \mathbf{B}_o = \begin{bmatrix} 0 & 1.4000 & 1 & 0 \\ 0.8878 & -1.7474 & 1.4100 & 0.5387 \end{bmatrix}$$

$$\mathbf{C}_o = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \; ; \; \mathbf{D}_o = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

The *Matlab* file used to design the observer, "roll_obs_des", is included in Appendix A.

VI. 21UUV COMPUTER MODEL AND SIMULATION

A. MODEL OVERVIEW

This chapter describes the 21UUV model and how inputs were made to produce desired simulations. Modifications were made to the original model designed by Healey and Miguel to accommodate specific simulations. The model is a combination of *Simulink* and *Matlab* files. *Matlab* files used in the model are in Appendix B. Figure 4.1 shows an overview of the 21UUV computer model.

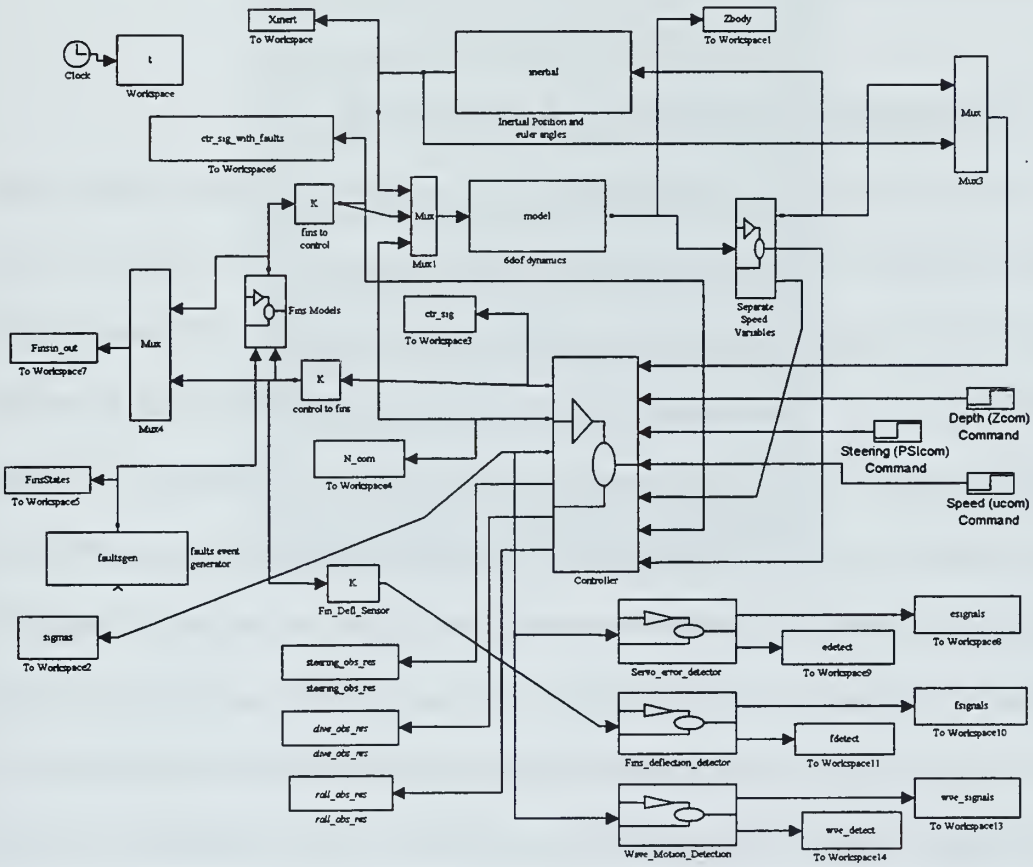

Figure 4.1 21UUV Computer Model

Simulations were run using the *Simulink* file "sim_uuv." The command "start_up" runs an *Matlab* file that sets up the defined set of hydrodynamic coefficients in the stack which are necessary and used as global variables for the simulation. After a simulation is conducted, the *Matlab* file "disp_res(Xinert,Zbody,t)" plots the overall results for the model. This includes all the velocities, translational and rotational variables. The model can be divided into seven sections: six degree of freedom dynamics, inertial position and euler angles, commands to the vehicle, sliding mode controllers, fault detectors, faults event generator, and fins model. Each system will be described in subsequent sections.

B. SIX DEGREE OF FREEDOM DYNAMIC MODEL

The heart of the simulation program is a six degree of freedom dynamic model using hydrodynamic coefficients and math models from the Naval Undersea Warfare Center (NUWC) report *Hydrodynamic Coefficients and Six Degree of Freedom Model for the NUWC UUV.* These identify inertial, lift, drag, and added mass parameters. Standard equations of motion were employed, but configured to include an "X" rather than a cruciform stern plane configuration. Unreported work by Marteno (1997) as a summer project solved the equations of motion using *Maple*, then created a ".cmex" file using the 'C' compiler in a "unix" platform. The ".cmex" files, appropriate for solving for vehicle body frame based inertial and relative velocities, were included in the S-function called "model", while the S-function "inertial" integrates velocities into global frame positions. Simulated wave and current programs are also included in the model. Other *Matlab* files in this section include "body_vel", "crossflow", and "wavevel."

Inputs were made into the "6dof dynamics" block. Figure 4.2 shows the "6dof dynamics" input block. "u0" is the initial surge, "v0" is the initial sway, "w0" is the initial heave, "p0" is the initial roll rate, "q0" is the initial pitch rate, "r0" is the initial yaw rate, and "amp" is the wave amplitude. In the case shown in the figure, the initial surge is 6 feet per second and the wave amplitude is 2 feet. All other variables are zero.

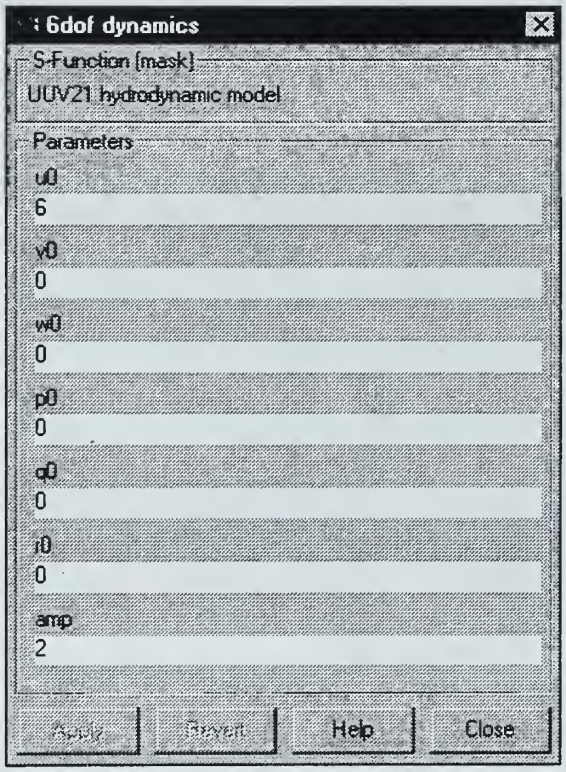

Figure 4.2 Six Degree of Freedom Dynamics Input Block

C. INERTIAL POSITION AND EULER ANGLES

The inertial position and euler angle model also used well known transformations to link body frame velocities to inertial positions. This model produced the translational

and rotational relationships needed to properly simulate the vehicle. Relevant *Matlab* files in this section include "inertial", "inertial_eq", "euler2body6d", and "predict."

Inputs were made into the "Inertial Position and Euler Angles" block. Figure 4.3 shows the "Inertial Position and Euler Angles" input block. All inputs are made into the "S-function parameters" line separated by commas. The first three numbers are the initial translational relationships X, Y, Z. The next three relationships, are the initial euler angles φ, θ, and ψ. In the case shown in the figure, the initial Z or depth is 10 feet. All other variables are zero.

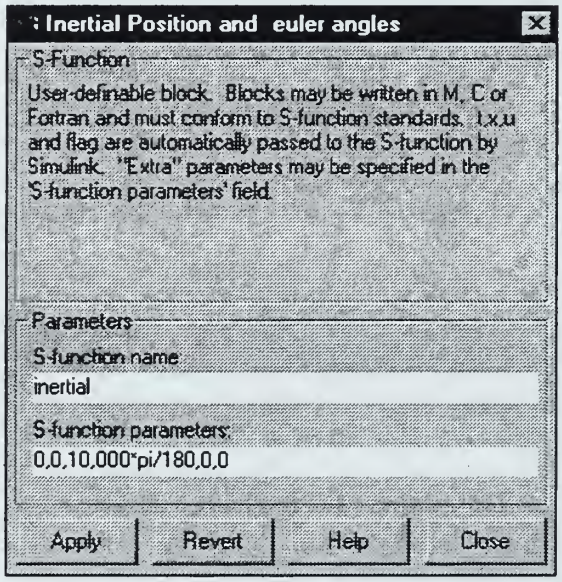

Figure 4.3 Inertial Position and Euler Angle Input Block

D. COMMANDS TO THE VEHICLE

Commands to the vehicle can also be altered during the simulation. The three commands depth, steering and speed are altered using a step input.

1. Depth

The depth command can be altered during a simulation. Figure 4.4 shows the "Depth (Zcom) Command" input block. The first line "Step time" indicates the start time for the vehicle to perform the maneuver. The second line "Initial value" is the initial depth of the vehicle. This input should be the same as the third entry in the "Inertial Position and Euler Angles" block. The third line "Final value" is the new commanded depth. In the case shown in the figure, at 30 seconds the vehicle is commanded to dive to a final value of 20 feet from an initial value of 10 feet.

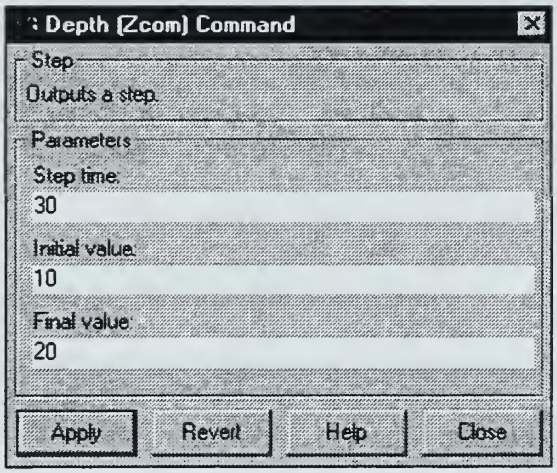

Figure 4.4 Depth (Zcom) Command Input Block

2. Steering

The steering command can be altered during a simulation. Figure 4.5 shows the "Steering (PSIcom) Command" input block.

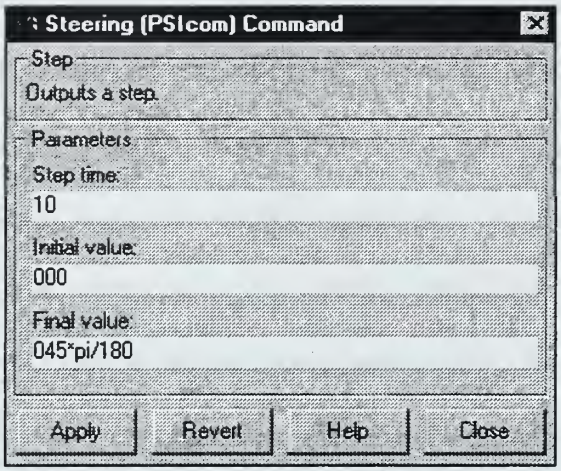

Figure 4.5 Steering (PSIcom) Command Input Block

The first line "Step time" indicates the start time for the vehicle to perform the maneuver. The second line "Initial value" is the initial course of the vehicle. This input should be the same as the fourth entry in the "Inertial Position and Euler Angles" block. The third line "Final value" is the new commanded course. In the case shown in the figure, at 10 seconds the vehicle is commanded to steer to a final course of 045 degrees from an initial course of 000 degrees.

3. Speed

The speed command can be altered during a simulation. Figure 4.6 shows the "Speed (ucom) Command" input block. The first line "Step time" indicates the start time for the vehicle to perform the speed change. The second line "Initial value" is the initial speed of the vehicle. This input should be the same as the "u0" entry in the "6dof dynamics" block. The third line "Final value" is the new commanded speed. In the case

shown in the figure, at 40 seconds the vehicle is commanded to increase speed to 10 feet per second from an initial speed of 6 feet per second.

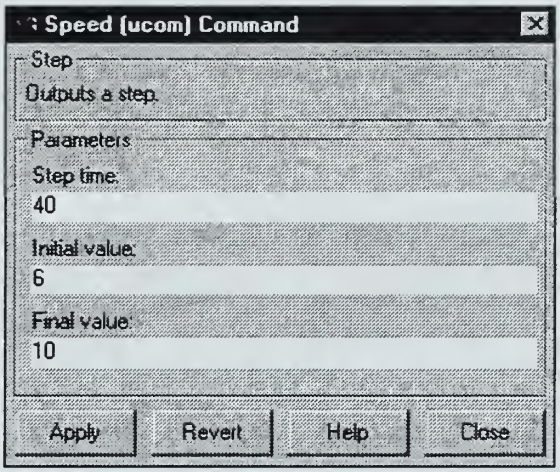

Figure 4.6 Speed (ucom) Command Input Block

E. SLIDING MODE CONTROLLERS

The sliding mode controllers for the model processes inputs from the six degree of freedom model, the inertial position and euler angle model, and maneuvering commands and passes the appropriate signal to the fins for action. Figure 4.7 shows a block diagram for the control system.

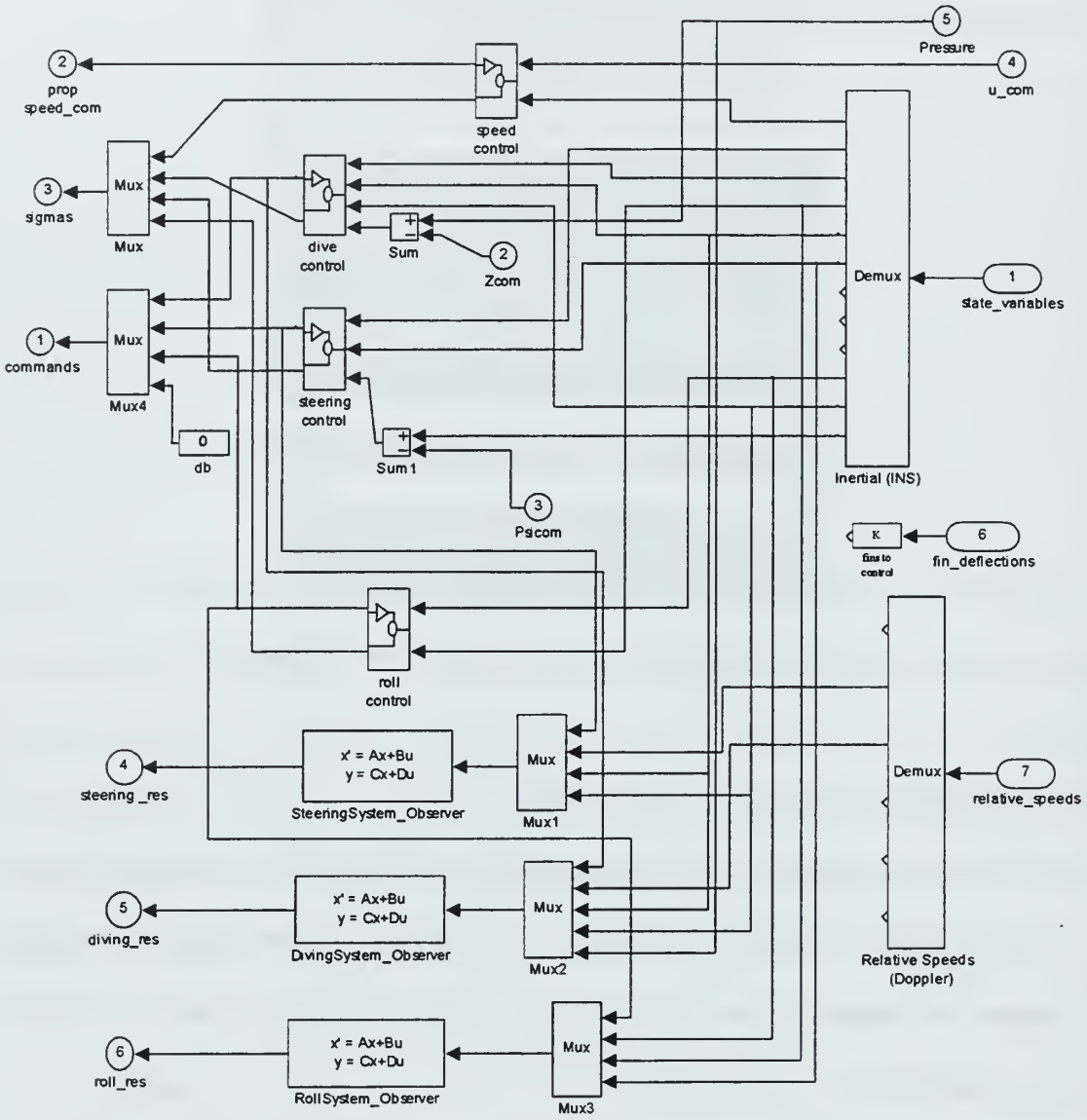

Figure 4.7 21UUV Computer Model

Relevant *Matlab* files in this section include "control_design", "ctr_dive", and "ctr_steering." There are four different controllers: diving, steering, roll, and speed.

1. Diving

The input for the dive control system is shown in Figure 4.8. The first two inputs, nonlinear gain and layer thickness, were derived from empirical data. The gain matrix (K), the s vector, and the B vector were all obtained from the *Matlab* file 'control_design.'

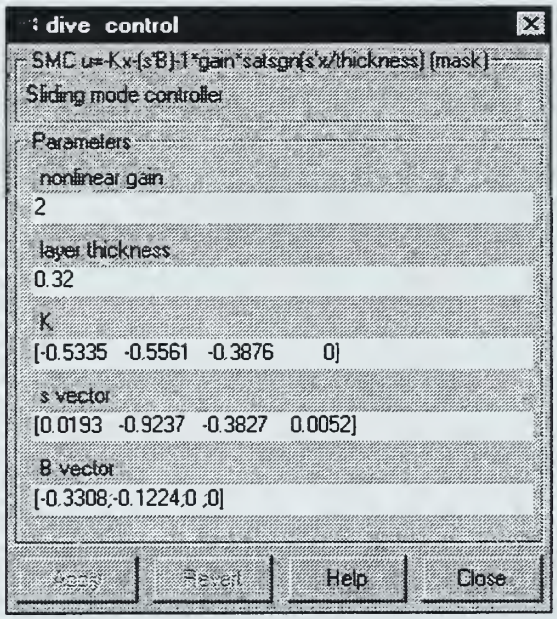

Figure 4.8 Dive Control Input Block

2. Steering

The input for the steering control system is shown in Figure 4.9. The first two inputs, nonlinear gain and layer thickness, were derived from empirical data. The gain matrix (K), the s vector, and the B vector were all obtained from the *Matlab* file "control_design."

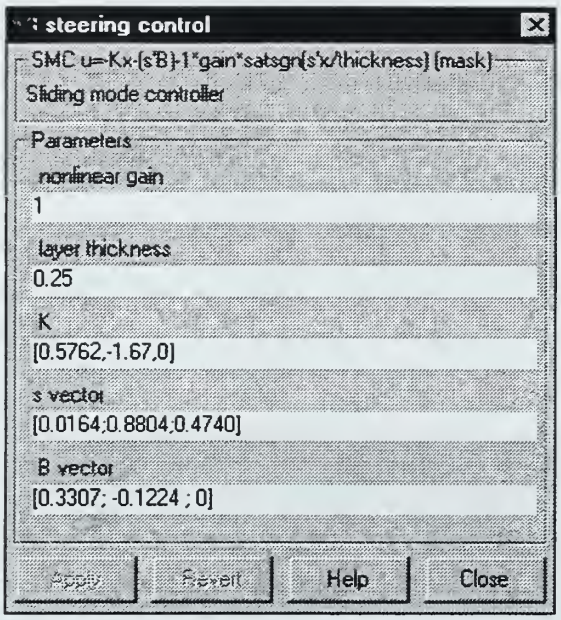

Figure 4.9 Steering Control Input Block

3. Roll

The input for the steering control system is shown in Figure 4.10.

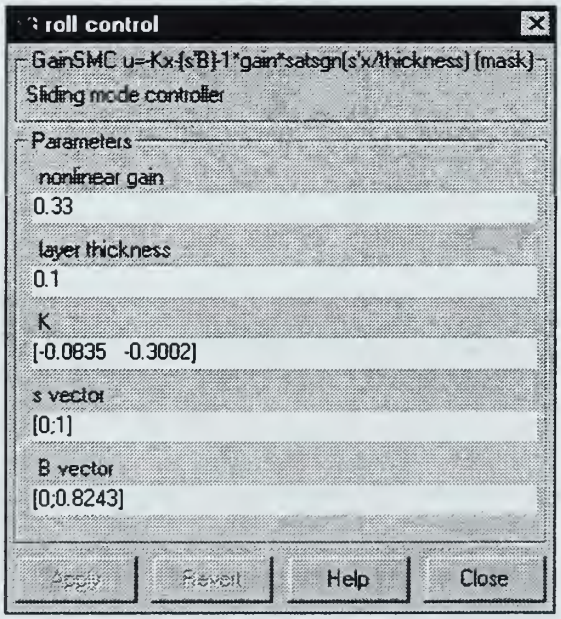

Figure 4.10 Roll Control Input Block

The first two inputs, nonlinear gain and layer thickness, were derived from empirical data. The gain matrix (K), the s vector, and the B vector were all obtained from the *Matlab* file "control_design."

4. Speed

The speed control system is shown in Figure 4.11 as a block diagram.

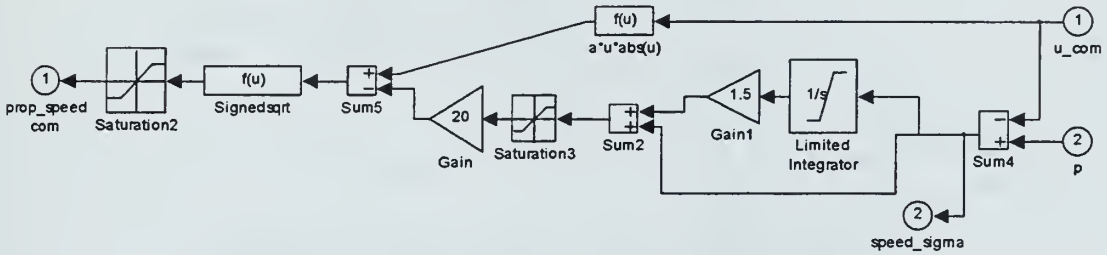

Figure 4.11 Speed Control System

F. FAULT DETECTORS

The fault detectors on the model take inputs from the output of the controller, compare that value with a threshold value, and determine whether a fault has occurred. The detector passes two signals, the original and a fault/no fault signal, for future implementation into a higher level control.

Four types of fault detectors are used in the model: servo error, fins deflection, wave motion, and observer residuals.

1. Servo Error

The servo error detector takes the "sigma" output from each of the four controllers, compares that value with a threshold value, and determines whether a fault has occurred. The two signals passed on for future implementation to a higher level control are "esignals" and "edetect." The *Matlab* file "disp_signals" plots the results from this detector. Figure 4.12 displays the servo error detector block diagram.

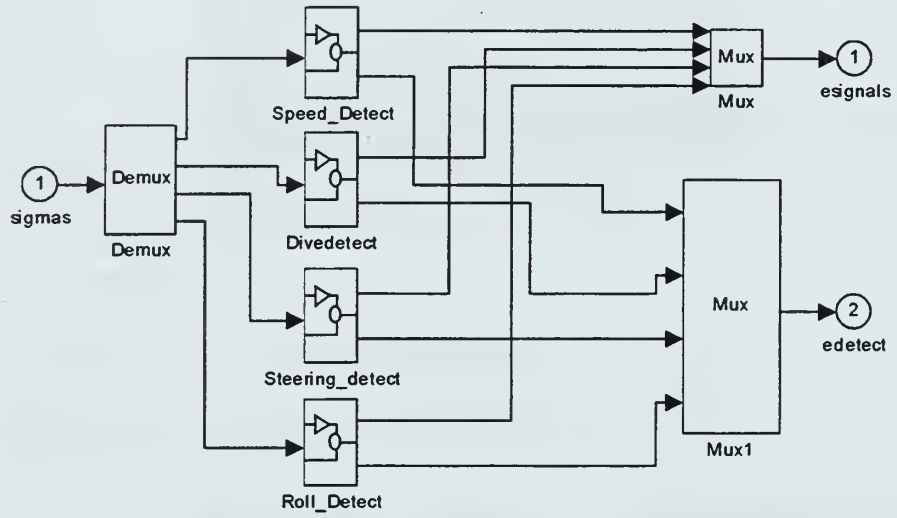

Figure 4.12 Servo Error Detector Block Diagram

2. Fins Deflection

The fin deflection detector takes the "commands" from each of the four controllers and converts the outputs into a fins deflection sensor by running through two gain matrices. That output is compared with a threshold value to determines whether a fault has occurred. The two signals passed on for future implementation to a higher level control are "fsignals" and "fdetect." The *Matlab* file "disp_f" plots the results from this detector. Figure 4.13 displays the fins deflection detector block diagram.

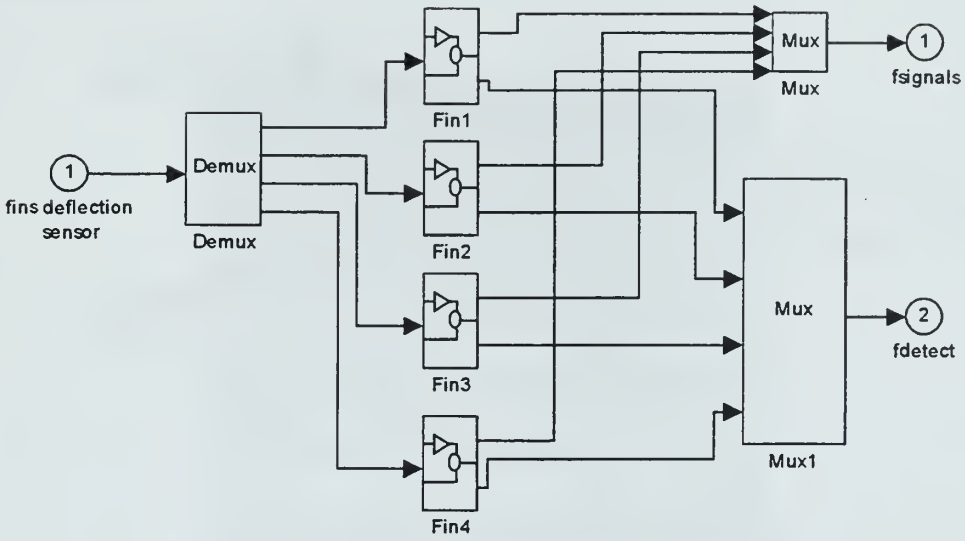

Figure 4.13 Fins Deflection Detector Block Diagram

3. Wave Motion

The wave motion detector is necessary for the model due to the near surface operation of future UUVs. The wave motion detector takes the "sigma" output from each of the four controllers, compares that value with a threshold value, and determines whether a fault has occurred. The two signals passed on for future implementation to a higher level control are "wve_signals" and "wve_detect." The *Matlab* file "disp_signals" plots the results from this detector. Figure 4.14 displays the wave motion detector block diagram.

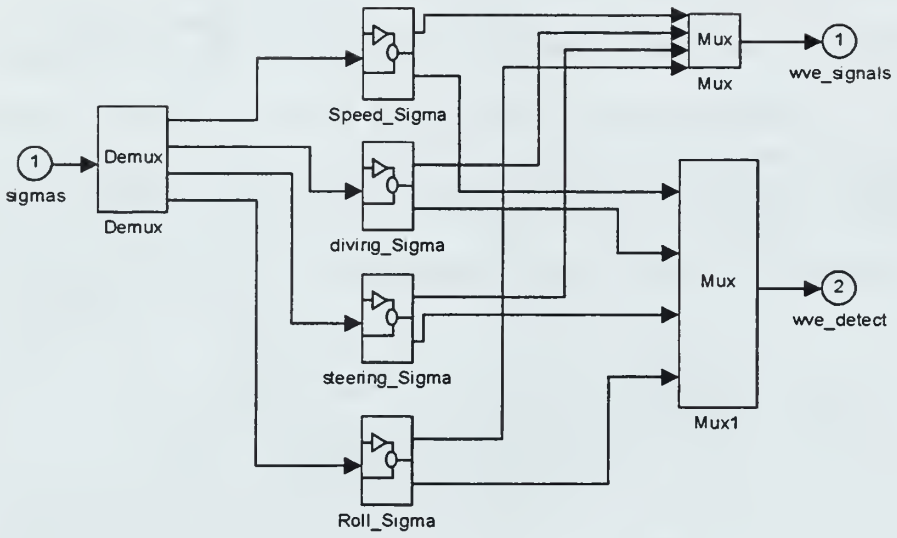

Figure 4.14 Wave Motion Detector Block Diagram

4. Observer Residual

The design of observer residual fault detectors is discussed in depth in Chapter III. The purpose of this subsection is to describe how inputs are made into each of the three observers: diving, steering and roll.

a. Diving

Figure 4.15 shows the "Diving System Observer" input block. The A, B, C and D matrices, as described in Chapter III, were obtained from the *Matlab* file "dive_obs_des." Commas separate all inputs made to the "Initial Conditions" line. The four entries are w_r, the heave velocity of the vehicle relative to the water, q, the pitch rate, θ, the pitch angle and Z, the depth. In this case shown in the figure, the initial depth is 10

feet and all other variables are zero. The *Matlab* file "disp_obs_dive" plots the results from this detector.

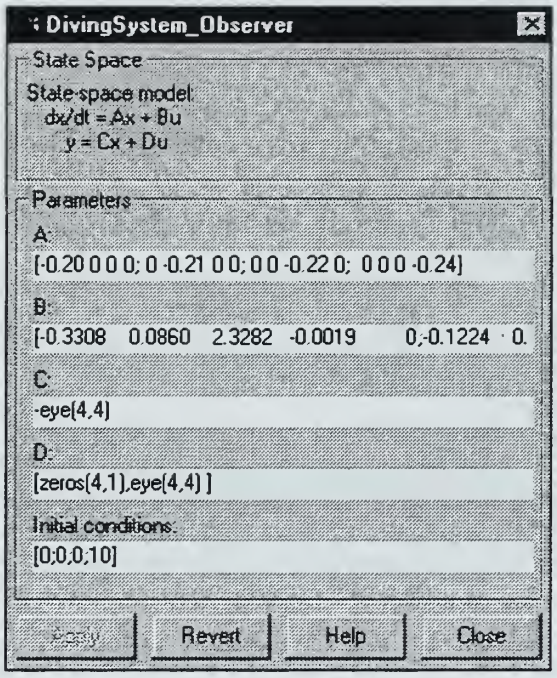

Figure 4.15 Diving System Observer Input Block

b. Steering

Figure 4.16 shows the "Steering System Observer" input block. The A, B, C and D matrices, as described in Chapter III, were obtained from the *Matlab* file "steer_obs_des." Commas separate all inputs made to the "Initial Conditions" line. The three entries are v_r, the sway velocity of the vehicle relative to the water, r, the yaw rate, and ψ, the yaw angle. In this case shown in the figure, the initial depth all variables are zero. The *Matlab* file "disp_obs_steer" plots the results from this detector.

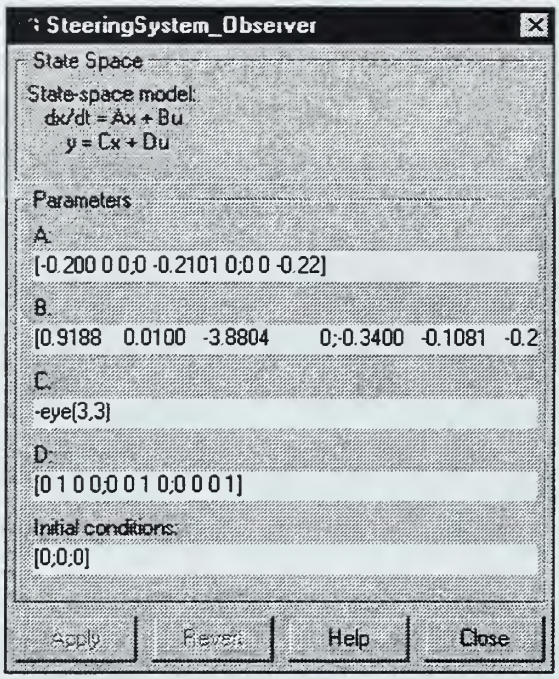

Figure 4.16 Steering System Observer Input Block

c. Roll

Figure 4.17 shows the "Roll System Observer" input block. The A, B, C, and D matrices, as described in Chapter III, were obtained from the *Matlab* file "roll_obs_des." Commas separate all inputs made to the "Initial Conditions" line. The two entries are p, the roll rate, and φ, the roll angle. In this case shown in the figure, all variables are zero. The *Matlab* file "disp_obs_roll" plots the results from this detector.

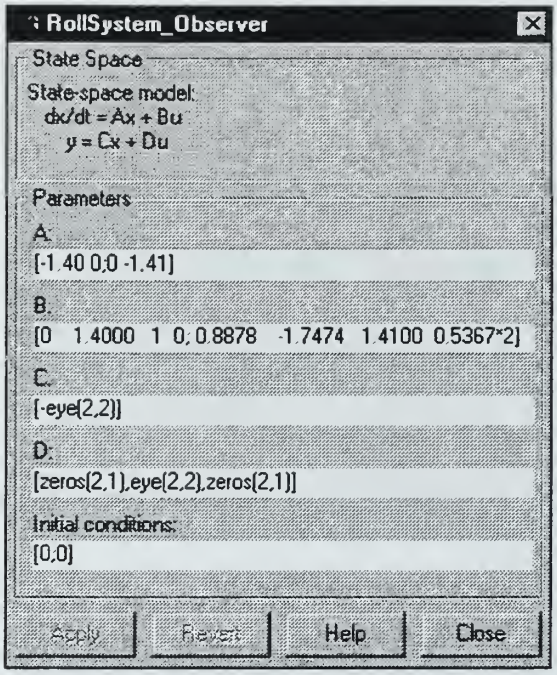

Figure 4.17 Roll System Observer Input Block

G. FAULTS EVENT GENERATOR

The faults event generator can change of the status of any of the four fins on the model at any time. Figure 4.18 shows the "Faults event generator" input block. The first line "S-function name" refers to the *Matlab* file "faultsgen." This file was used to create the faults event generator. The second line "S-function parameters" has two matrix inputs. The first is a four by one matrix, which indicates the initial condition of the fins. Each element in this matrix represents a fin. The first element is fin 1, the second fin 2, etc. The numbers for each element indicate a specific "condition." "1" indicates the fin is in normal operation, "2" indicates that the fin is stroke limited (can only move to 0.25 radians instead of the full 0.4), "3" indicates that the fin is loose (ineffective, stays at

zero), and "4" indicates that the fin is stuck (stays at 0.4 radians). In the case shown in the figure, all four fins are initially operating normally.

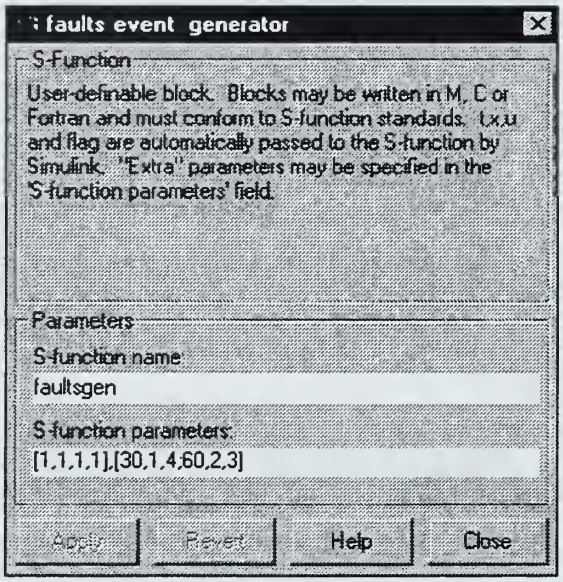

Figure 4.18 Faults Event Generator Input Block

The second input is a three column matrix. The first column represents the time of the action, the second column represents the fin number, and the third column represents the "condition" of the fin. This matrix can have a maximum of six rows. In the case shown in the figure, at 30 seconds a stuck rudder was imposed to fin 1 and at 60 seconds a loose rudder was imposed to fin 2.

H. FINS MODEL

The process of how a fault is added to the model is shown in Figure 4.19.

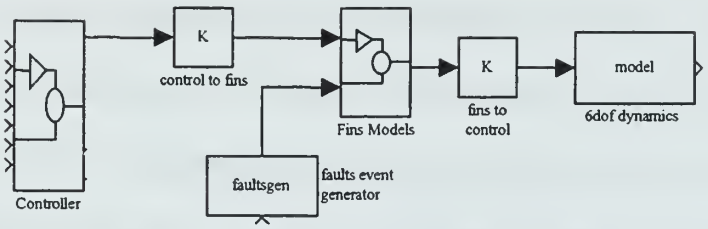

Figure 4.19 Fault Addition Process

Commands from the diving, steering, and roll control systems are inputted in the "control to fins" block. In this block, the commands from each of the control systems are multiplied by a gain matrix and converted into deflections for each fin (δ_1, δ_2, δ_3, δ_4).

The fins model takes inputs from the fin deflections and faults events generator and adds the signals together to produce δ_{f1}, δ_{f2}, δ_{f3}, and δ_{f4}. Figure 4.20 shows a block diagram of the fin model.

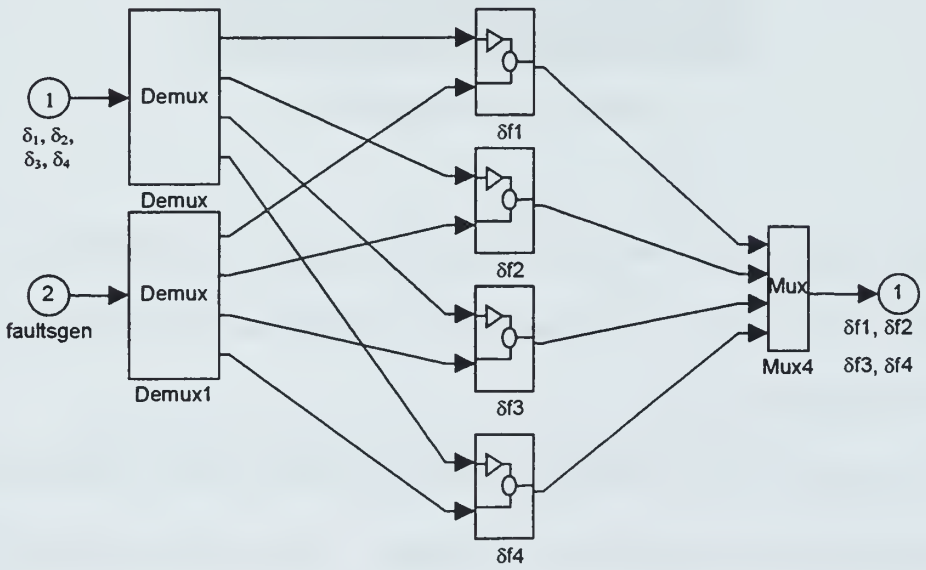

Figure 4.20 Fins Model Block Diagram

By entering the fin subsystem, parameters can be changed for each fin. Figure 4.21 shows the "fin1" input block. The first line "Max fin stroke" refers to the maximum turning capability of the fin. The second line "limited fin stroke [fault type 2]" refers to the maximum turning capability when at stroke limited condition. The third line "stuck fin [fault type 4]" refers to the position of the fin when at a stuck fin condition. In the case shown in the figure, the maximum fin stroke is 0.4 radians, the limited fin stroke condition is 0.25 radians, and the stuck fin condition is at 0.4 radians.

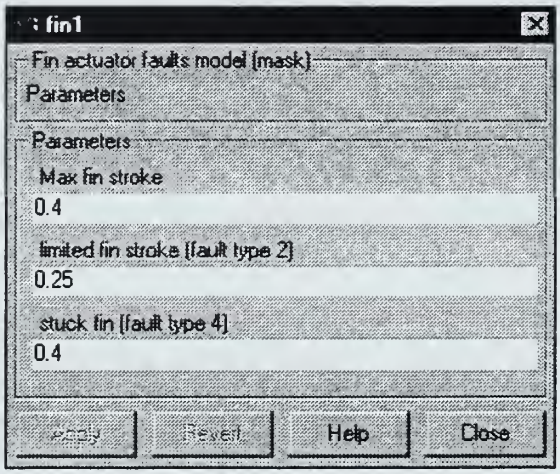

Figure 4.21 Fin Input Block

After exiting the fins model block, δ_{f1}, δ_{f2}, δ_{f3}, and δ_{f4} are multiplied by another gain matrix, "fins to control", to produce the dive, steering, and roll commands for input into the 6dof dynamics model.

An example of a dive command, δ_s, is shown in Figure 4.22. The four fins in an "X" configuration are shown from a stern aspect. The arrows represent the force

direction of each respective fin. In this case, all horizontal forces cancel out and the remaining forces add up to produce an upward deflection.

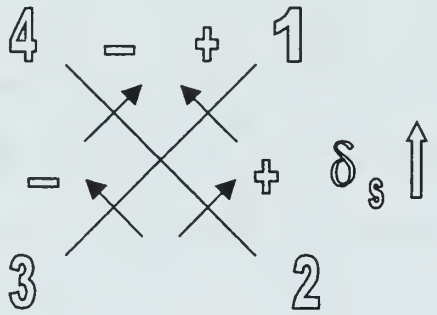

Figure 4.22 Dive Command, δ_s, Example (Stern View)

Figure 4.23 shows an example of a steering command, δ_r. In this case, all vertical forces cancel out and the remaining forces add up to produce a deflection to the right.

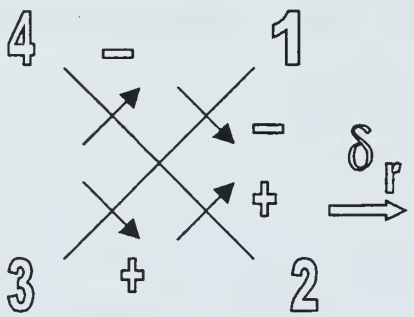

Figure 4.23 Steering Command, δ_r, Example (Stern View)

Figure 4.24 shows an example of a roll command, δ_a. In this case, all horizontal and vertical forces are canceled out but a clockwise moment is produced.

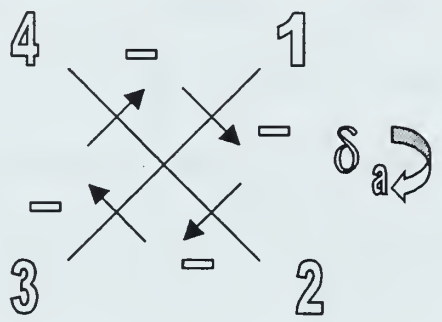

Figure 4.24 Roll Command, δ_a, Example (Stern View)

V. SIMULATION RESULTS FOR SPECIFIC CASES

This study uses the 21UUV computer model to simulate actions of an actual autonomous underwater vehicle in shallow water operations. To test the effectiveness of model based observers for fault detection, five specific areas of study were chosen: robustness of "X" fin configuration, weight and buoyancy mismatch, detection of fin faults in the presence of waves, fin fault detection using maximum likelihood analysis, and control at slow speeds.

A. ROBUSTNESS OF "X" FIN CONFIGURATION

This study was conducted to determine if the "X" fin configuration of 21UUV provides a level of redundancy, which the control systems (diving, steering, roll, and speed) can use to reconfigure the autopilot system. In the 21UUV model, fin 1 is redundant with fin 3 and fin 2 is redundant with fin 4.

Scenario 1: fin fault followed by steering command with weight mismatch. In the first scenario, the vehicle is moving forward at a speed of 8 feet per second. The depth of the water column is 30 feet and the vehicle is at 10 feet. The vehicle is on a course of 000. For this simulation, there are no waves. 20 seconds into the simulation, a stuck #1 fin fault was imposed with a level of 0.4 radians. At 50 seconds, the vehicle is commanded to steer to 045. The simulation was run for 100 seconds. Figure 5.1 shows an "X-Y" plot of the vehicle's path trajectory. This plot shows that despite a fault to fin 1, the vehicle was successfully able to turn to 045, and crudely maintaining heading in spite of the fault.

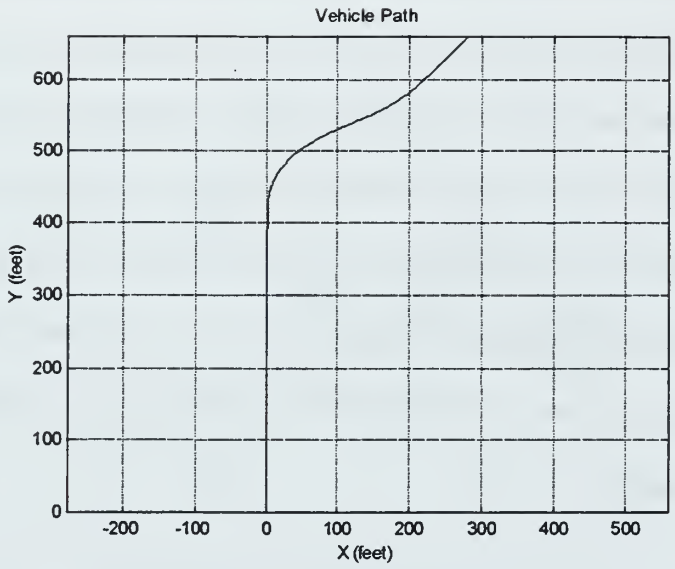

Figure 5.1 Vehicle Path for Scenario 1

A closer examination of the turn can be determined by the yaw angle, ψ. Figure 5.2 shows that the vehicle was able to maintain a course of 000 after the fin fault occurred at 20 seconds.

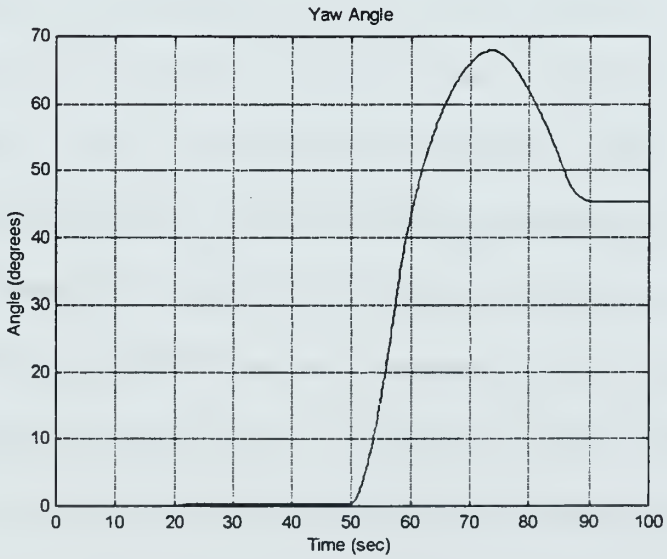

Figure 5.2 Yaw Angle for Scenario 1

When the turn was commanded at 50 seconds, the vehicle turned and steadied up on 045 by 85 seconds. Note that the vehicle actually turned to 065 at 75 seconds before reaching a steady state.

To see how fin 3 compensated for the fault to fin 1, a plot for fin response versus time is examined. Figure 5.3 shows the fin response for fins 1 and 3. The response of the fins is equal in magnitude until the fault occurs at 20 seconds. Then, fin 3 adjusts quickly to compensate for the fin 1 stuck condition. When the turn is executed at 50 seconds, fin 3 adjusts again to compensate for fin 1, although control dynamics are degraded. Therefore, it appears that one fin can be lost on 21UUV without compromising the steering capability of the vehicle.

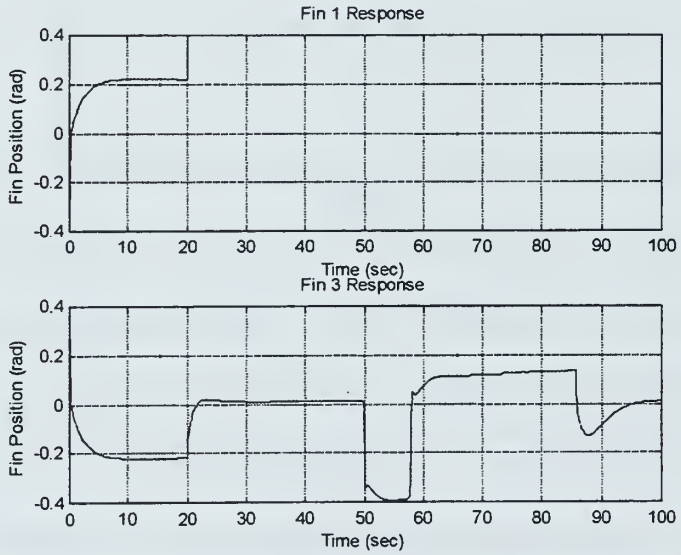

Figure 5.3 Fins 1 and 3 Response for Scenario 1

To determine the fault detection capability of the observers, the steering and roll residuals will be examined. For the steering observer, the residuals of interest are the sway velocity of the vehicle relative to the water, v_r, (sensed by acoustic doppler) and, the yaw rate, r. Figure 5.4 shows the steering observer residual response for v_r and r. Note that both residuals show a response for the fin fault at 20 seconds, but both residuals also show a response of equal or greater magnitude for the turn at 50 seconds. The steering observer residuals can not effectively distinguish a fault from a turn. This is due to the fact that fin faults are additive in the steering control system. Thus, steering observer residuals are seen to be responsive to maneuvering conditions, in spite of simple theory, and may only be used to detect faults during nonmanuevering conditions.

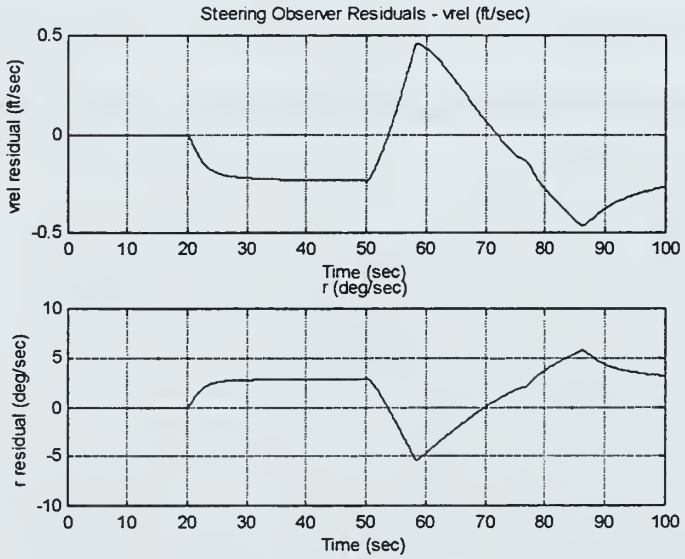

Figure 5.4 Steering Observer Residuals v_r and r for Scenario 1

The roll observer residual of interest is the roll rate, p. Figure 5.5 shows the roll observer residual response for p. Note that the residual shows an immediate response for

the fin fault at 20 seconds. The magnitude of response for the turn at 50 seconds is less than for the fin fault. Although fin faults are additive in the roll control system, the roll command is always set back to zero.

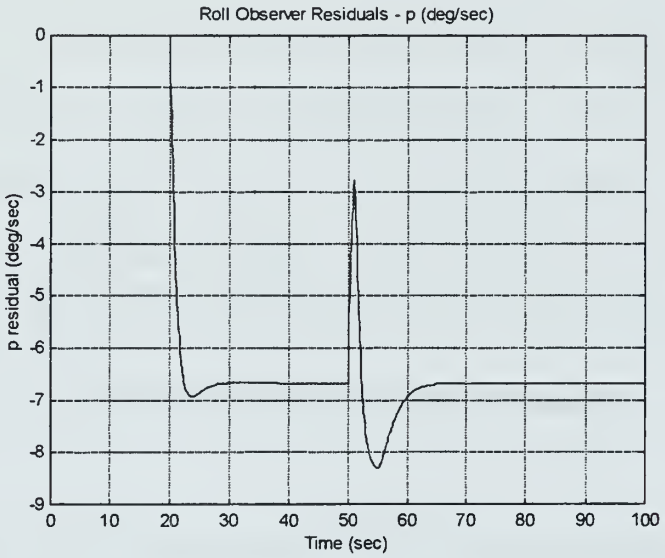

Figure 5.5 Roll Observer Residual *p* for Scenario 1

Scenario 2: fin fault followed by diving command with weight mismatch. In the second scenario, the vehicle is moving forward at a speed of 8 feet per second. The depth of the water column is 30 feet and the vehicle is at 25 feet. The vehicle is on a course of 000. For this simulation, there are no waves. 20 seconds into the simulation, a stuck #2 fin fault was imposed. At 40 seconds, the vehicle is commanded to "dive" to 0 (surface). The simulation was run for 100 seconds. Figure 5.6 shows a depth plot of the vehicle. This plot shows that despite a fault to fin 2, the vehicle was successfully able to surface.

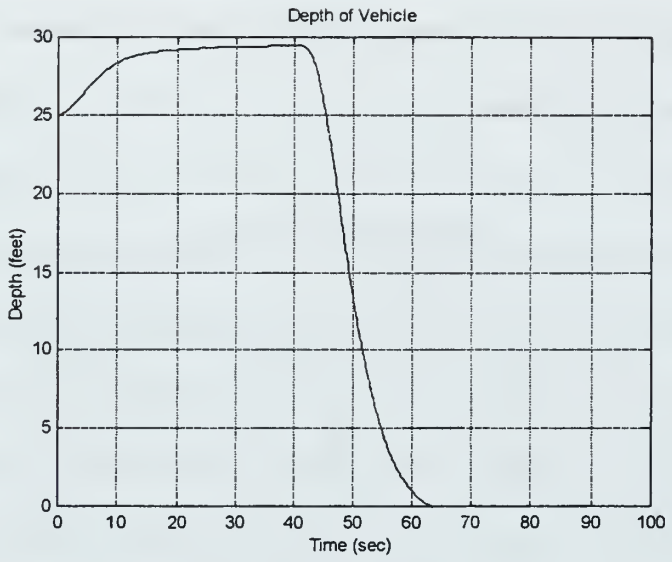

Figure 5.6 Depth of Vehicle for Scenario 2

A closer examination of the "dive" can be determined by the pitch angle, θ. Figure 5.7 shows the pitch angle versus time.

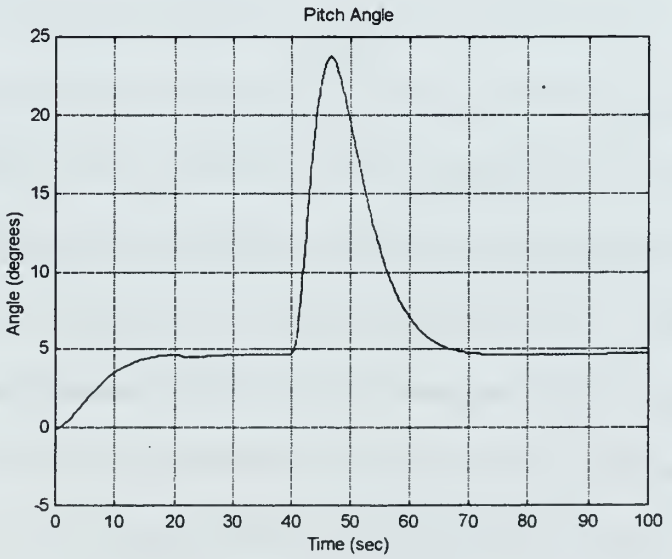

Figure 5.7 Pitch Angle for Scenario 2

The vehicle was able to maintain depth at 25 ft after the fin fault occurred at 20 seconds. When the "dive" was commanded at 40 seconds, the pitch angle increased to 25 degrees. After surfacing at 60 seconds, the vehicle returned to the steady state pitch angle before the maneuver.

To see how fin 4 compensated for the fault to fin 2, a plot for fin response versus time is examined. Figure 5.8 shows the fin response for fins 1 and 3. The response of the fins is equal in magnitude until the fault occurs at 20 seconds. Then, fin 4 adjusts quickly to compensate for the fin 2 stuck condition. When the "dive" is executed at 40 seconds, fin 4 adjusts again to compensate for fin 2. Therefore, it appears that one fin can be lost on 21UUV without compromising the diving capability of the vehicle.

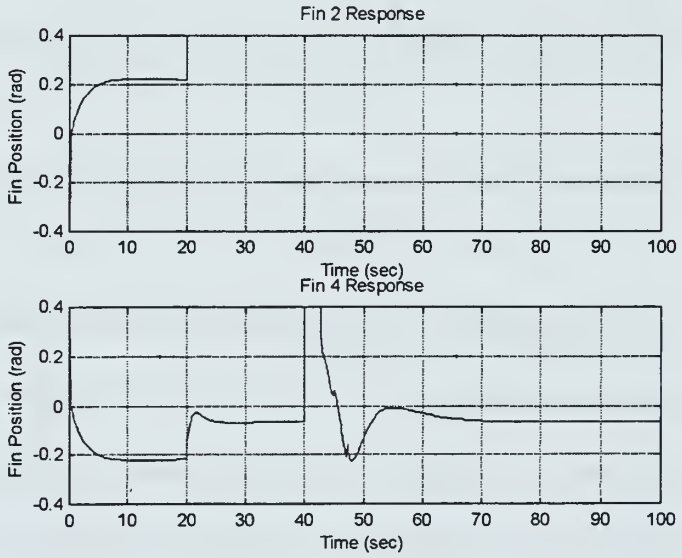

Figure 5.8 Fins 2 and 4 Response for Scenario 2

To determine the fault detection capability of the observers, the roll observer residuals were examined. The roll observer residual of interest is the roll rate, p. Figure

5.9 shows the roll observer residual response for p. Note that the residual shows an immediate response for the fin fault at 20 seconds. The magnitude of response for the "dive" at 40 seconds is less than for the fin fault. Thus, by setting an appropriate threshold, a fin fault can be distinguished from a maneuver.

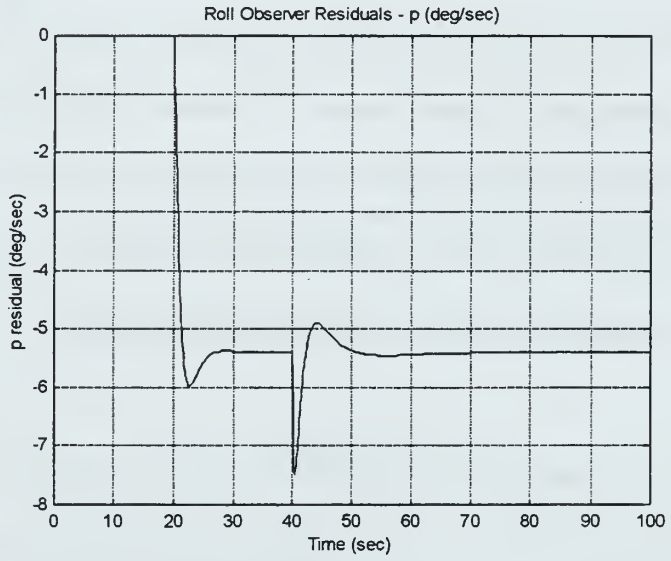

Figure 5.9 Roll Observer Residual p for Scenario 2

Scenario 3: low speed with two fin faults. In the third scenario, the vehicle is moving forward at a speed of 4 feet per second. The depth of the water column is 30 feet and the vehicle is at 10 feet. The vehicle is on a course of 000. For this simulation, there are no waves. 20 seconds into the simulation, a stuck #1 fin fault was imposed. At 40 seconds, a stuck #3 fin fault was imposed. The simulation was run for 80 seconds. Figure 5.10 shows a depth plot of the vehicle. This plot shows that after the second fin fault, the vehicle is unable to maintain depth control.

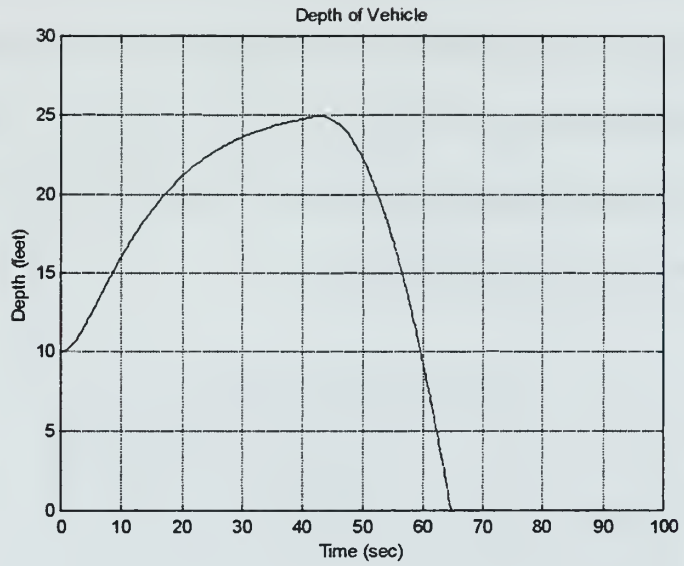

Figure 5.10 Depth of Vehicle for Scenario 3

Figure 5.11 shows the vehicle is also unable to maintain steering control after the second fault.

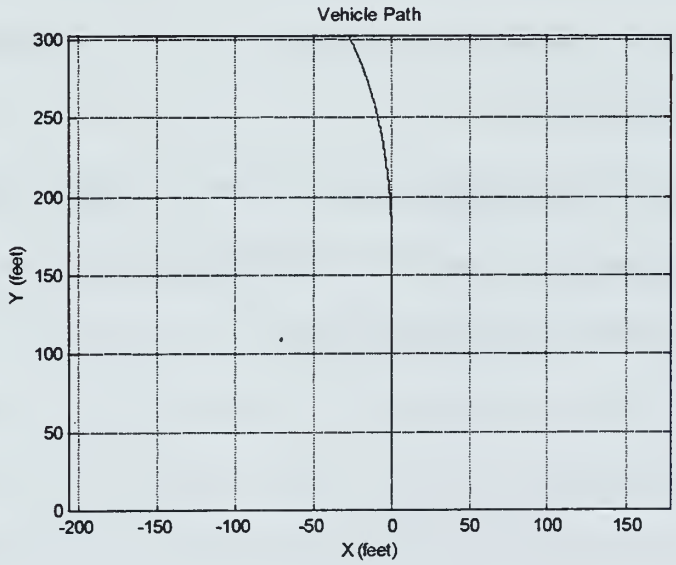

Figure 5.11 Vehicle Path for Scenario 3

To determine the fault detection capability of the observers, the steering and roll residuals were examined. For the steering observer, the residuals of interest are the sway velocity of the vehicle relative to the water, v_r, and, the yaw rate, r. Figure 5.12 shows the steering observer residual response for v_r and r. Note that both residuals show a huge response after the second fin fault at 40 seconds.

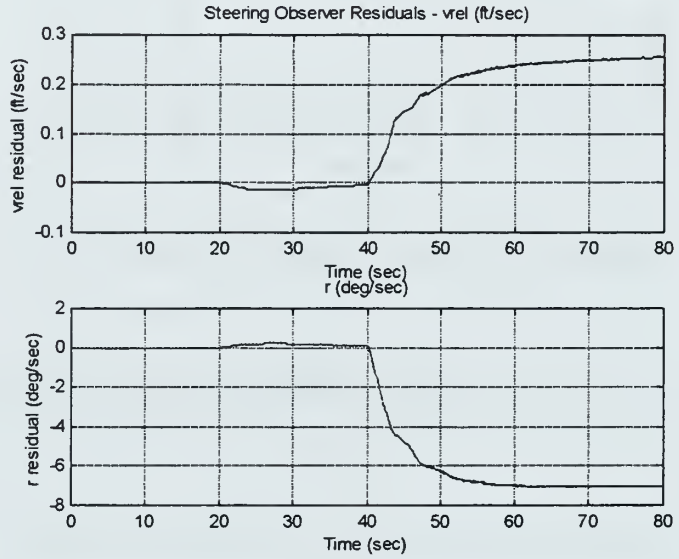

Figure 5.12 Steering Observer Residuals v_r and r for Scenario 3

The roll observer residual of interest is the roll rate, p. Figure 5.13 shows the roll observer residual response for p. Note that the residual shows a response after the first fin fault and a huge response after the second fin fault.

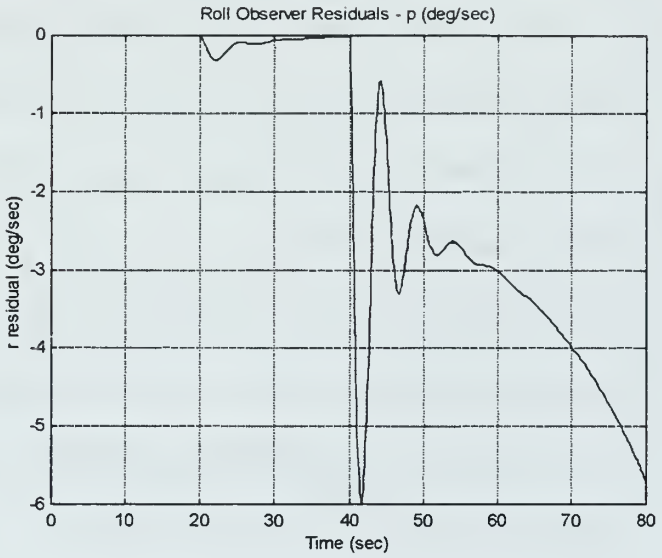

Figure 5.13 Roll Observer Residual p for Scenario 3

B. WEIGHT AND BUOYANCY MISMATCH

Due to the inherent difficulty of running the 21UUV with perfectly neutral balance, a study was conducted to determine if the diving observer residuals could detect a weight-buoyancy mismatch.

The first step in this study was to evaluate a vehicle where there was no weight-buoyancy mismatch (weight equals buoyancy). Scenario 4: Straight run with no weight-buoyancy mismatch. In the fourth scenario, the vehicle is moving forward at a speed of 8 feet per second. The depth of the water column is 30 feet and the vehicle is at 10 feet. The vehicle is on a course of 000. For this simulation, there are no waves. The simulation was run for 50 seconds. Figure 5.14 shows a depth plot of the vehicle. As expected, this plot shows that the vehicle is able to hold depth when there is no weight-buoyancy mismatch.

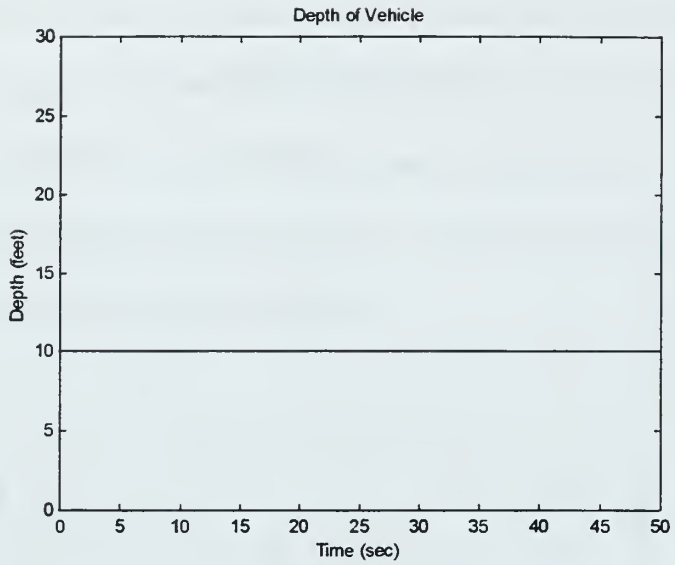

Figure 5.14 Depth of Vehicle for Scenario 4

When examining a model based fault detector, the diving observer residuals of interest are the heave velocity of the vehicle relative to the water, w_r, and the pitch rate, q. Figure 5.15 shows the diving observer residuals w_r and q and confirms that there are no faults in this scenario.

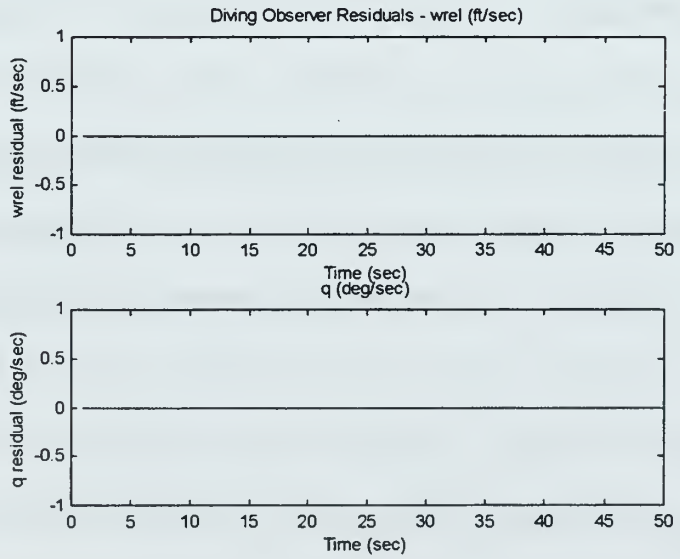

Figure 5.15 Diving Observer Residuals w_r and q for Scenario 4

The final step in this study was to evaluate a vehicle where there was weight-buoyancy mismatch (weight is 2 percent greater than buoyancy). Scenario 5: Straight run with weight-buoyancy mismatch. In the fifth scenario, the vehicle is moving forward at a speed of 8 feet per second. The depth of the water column is 30 feet and the vehicle is at 10 feet. The vehicle is on a course of 000. For this simulation, there are no waves. The simulation was run for 50 seconds. Figure 5.16 shows a depth plot of the vehicle. This plot shows that the vehicle is unable to hold the commanded depth of 10 feet. The vehicle sinks until reaching a steady state depth of 14 feet.

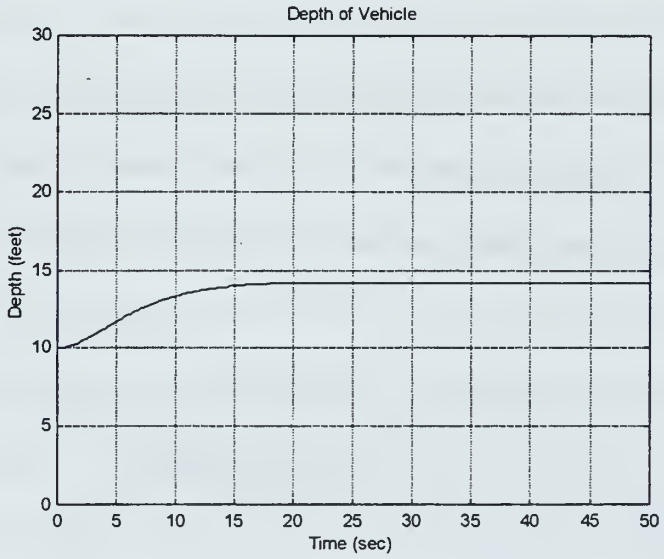

Figure 5.16 Depth of Vehicle for Scenario 5

A closer examination of the weight-buoyancy mismatch can be determined by the pitch angle, θ. Figure 5.17 shows the pitch angle for the scenario. Due to the excess weight of the vehicle, θ increased to 5 degrees and stayed there when the vehicle reached steady state.

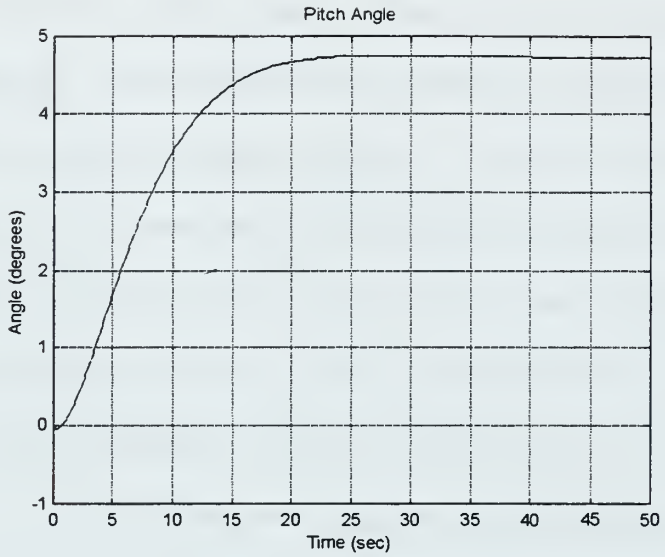

Figure 5.17 Pitch Angle for Scenario 5

When examining a model based fault detector, the diving observer residuals of interest are the heave velocity of the vehicle relative to the water, w_r, and the pitch rate, q. Figure 5.18 shows the diving observer residuals w_r and q for this scenario.

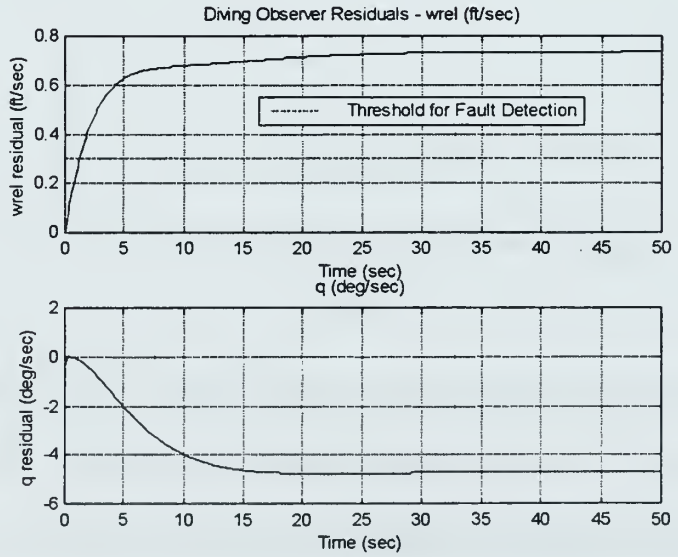

Figure 5.18 Diving Observer Residuals $\dot{w}_r$ and q for Scenario 5

As shown, the residual w_r immediately responds to the heaviness of the vehicle. The residual q responds a bit slower, but still detects the heaviness of the vehicle. Thus, it appears that w_r is more responsive than q. By setting a threshold value on w_r of 0.3 feet per second, any weight-buoyancy mismatch on the vehicle should be detected.

C. DETECTION OF FIN FAULTS IN THE PRESENCE OF WAVES

This study was conducted to determine if model based observers could be used to distinguish fin faults from wave disturbances. A properly designed model based observer should be unresponsive to wave disturbances, but should react to an actuator fault.

Scenario 6: fast speed with two fin faults and waves. In the sixth scenario, the vehicle is moving forward at a speed of 10 feet per second. The depth of the water column is 30 feet and the vehicle is at 10 feet. The vehicle is on a course of 000. For this simulation, the wave amplitude is set at 2 feet and the waves are coming from 000. 20 seconds into the simulation, a stuck #1 fin fault was imposed. At 60 seconds, a stuck #2 fin fault was imposed. The simulation was run for 100 seconds. Figure 5.19 shows a depth plot of the vehicle. This plot shows that the vehicle is affected by the wave disturbance. The first fin fault has little influence since the vehicle is able to reconfigure. The second fin fault causes the vehicle to lose depth control.

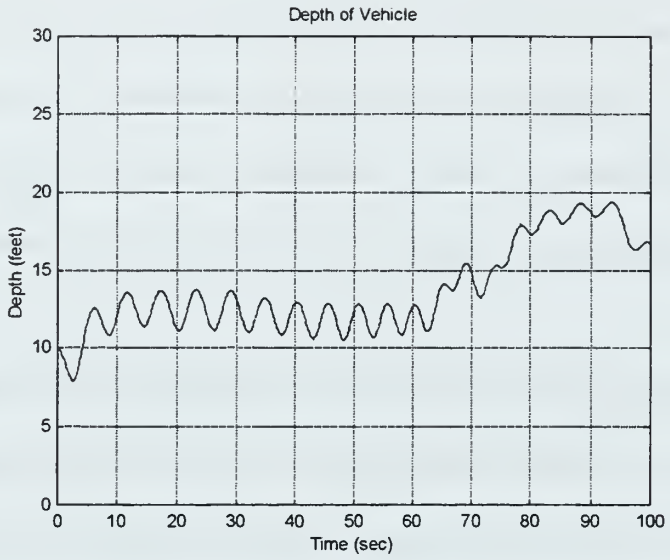

Figure 5.19 Depth of Vehicle for Scenario 6

Figure 5.20 shows the vehicle path. This plot shows that the vehicle loses steering control after the second fault occurs.

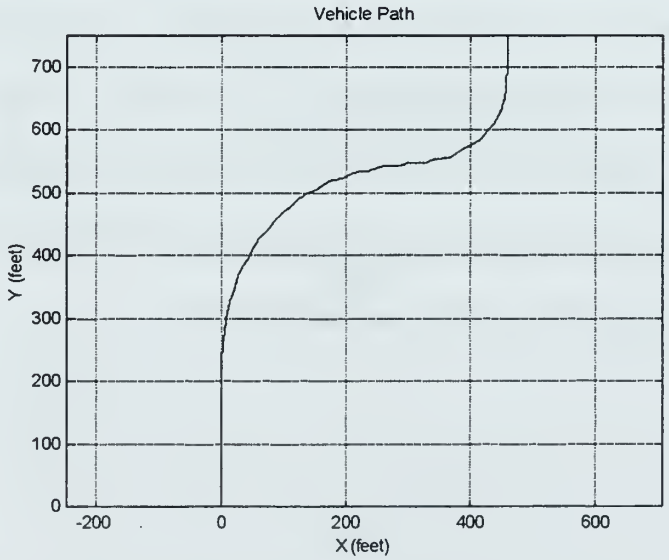

Figure 5.20 Vehicle Path for Scenario 6

To determine the fault detection capability of the observers, the roll observer residuals were examined. The roll observer residual of interest is the roll rate, p. Figure 5.21 shows the roll observer residual response for p. Note that the residual shows no response from 0 to 20 seconds. This means that the observer residual is not affected by wave disturbances. At 20 seconds, a response occurs during the fault to fin 1. At 60 seconds, an even bigger response occurs during the fault to fin 2. Therefore, by setting an appropriate threshold, a fin fault can be distinguished from wave disturbance. A threshold level of 3 degrees per second on p should detect a fin fault.

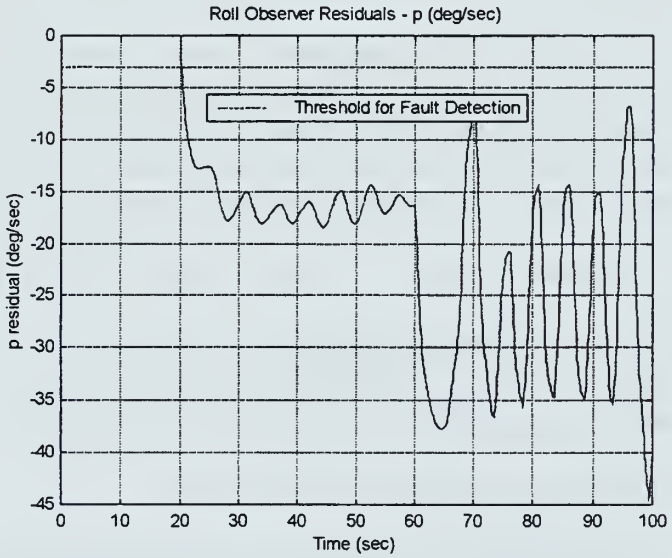

Figure 5.21 Roll Observer Residual p for Scenario 6

D. FIN FAULT DETECTION USING MAXIMUM LIKELIHOOD ANALYSIS

In order to more clearly distinguish faults from maneuvers in residual signals, likelihood functions are used to provide probability ratios. By cross correlating the roll

rate observer residual signal with an auto-correlated expected unit step fault response, the likelihood of a fault or maneuver is more clearly distinguishable. The likelihood function is described by Peng, et. al., (1997):

$$L(i) = \frac{2\max}{j}\left[\frac{1}{2}\frac{\left(\sum_{k=1}^{M-j+1}\rho_k v_{i-m+k+j-1}\right)^2}{\sum_{k=1}^{M-j+1}\rho_k^2}\right]$$

L(i) is the ratio of probability that v is caused by a step change to probability that v is caused by non-specific inputs. v_i is the residual signal at *i dt*. ρ_i is the unit step input response of the filter. M is defined as the smoothing window. j goes from 1 to M.

Scenario 7: medium speed run with steering command followed by a subsequent fin fault. In the seventh scenario, the vehicle is moving forward at a speed of 8 feet per second. The depth of the water column is 30 feet and the vehicle is at 10 feet. The vehicle is on a course of 000. For this simulation, there are no waves. 20 seconds into the simulation, the vehicle is commanded to steer to 045. At 60 seconds, a stuck #1 fin fault was imposed. The simulation was run for 100 seconds.

Figure 5.22 shows the plain roll rate observer residual signal for the scenario. The amplitude ratio between the fin fault and the steering maneuver is approximately 5 to 1.

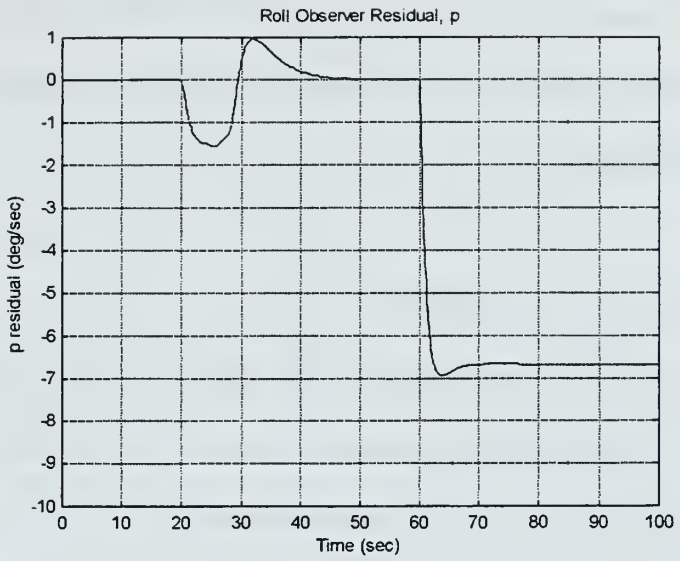

Figure 5.22 Roll Observer Residual, *p*, for Scenario 7

Figure 5.23 shows the roll rate observer residual signal after being run through a likelihood filter. The amplitude ratio between the fin fault and maneuvering response is now approximately 20 to 1.

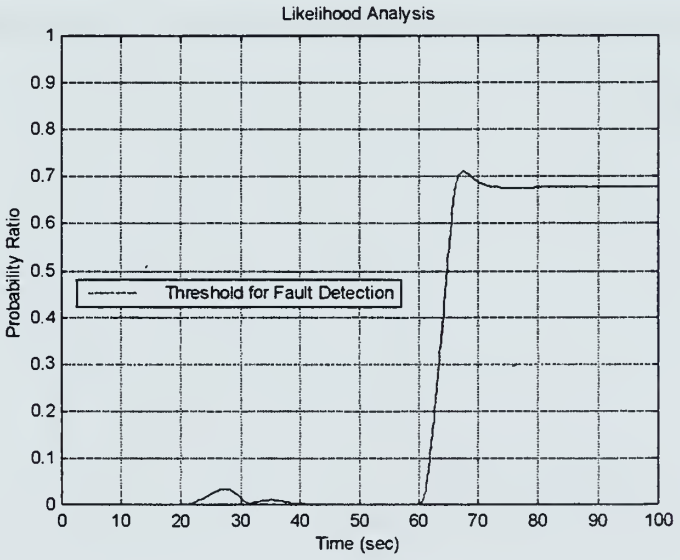

Figure 5.23 Maximum Likelihood Analysis on Roll Observer Residual, *p*

Thus, by using maximum likelihood analysis, the difference between a fin fault and maneuvering signal is clearly more distinguishable. A proposed threshold can be set for the detection of a fin fault at 0.3.

E. CONTROL AT SLOW SPEEDS

In the course of conducting simulations, the vehicle has shown degrading control as speed decreases. This study was conducted to determine why control problems arise when the vehicle is forced to go less than 6 feet per second.

Scenario 8: Slow speed run with command to steer with weight-buoyancy mismatch. In the eighth scenario, the vehicle is moving forward at a speed of 4 feet per second. The depth of the water column is 30 feet and the vehicle is at 10 feet. The vehicle is on a course of 000. For this simulation, the wave amplitude is set at 2 feet and the waves are coming from 000. At 20 seconds, the vehicle is commanded to steer to 045. The simulation was run for 80 seconds. Figure 5.24 shows an "X-Y" plot of the vehicle. This plot shows that the vehicle turns very slowly. At slow speeds, the vehicle is very slow to respond.

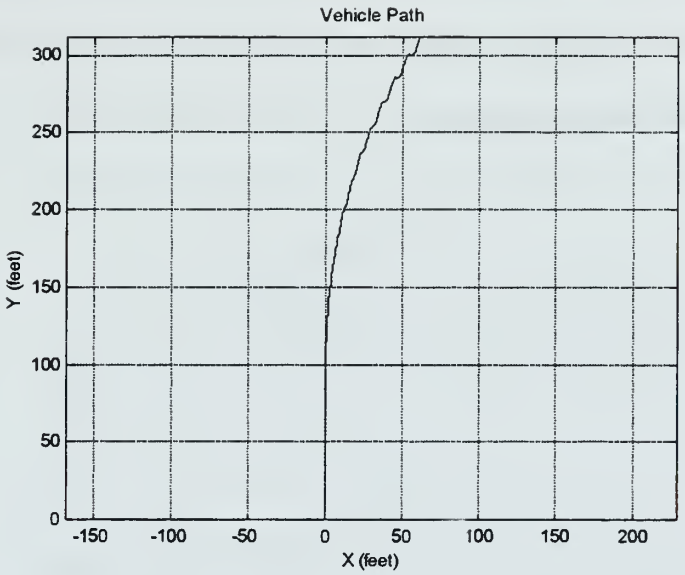

Figure 5.24 Vehicle Path for Scenario 8

Figure 5.25 shows a depth plot of the vehicle. This plot shows that the vehicle is unable to maintain depth when turning and actually surfaces at 67 seconds.

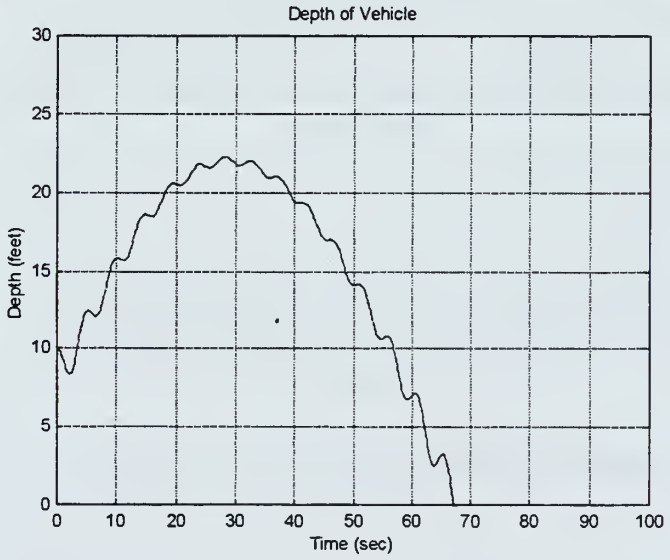

Figure 5.25 Depth of Vehicle for Scenario 8

Due to the lack of speed, reaction forces by the fins are not as great as at higher speeds. Figure 5.26 shows the fin response with respect to time for all four fins. Note that all four fins are near or at saturation.

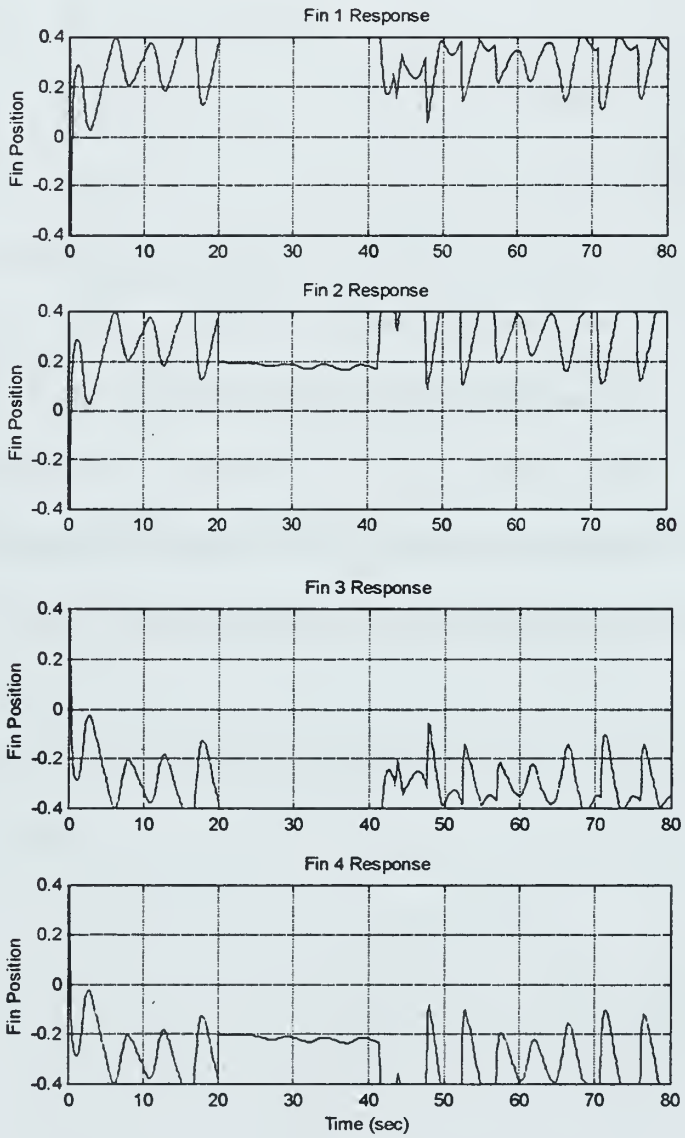

Figure 5.26 Response of all 4 fins in Scenario 8

Since the fins are forced to use so much of their authority to maintain depth at slower speeds, a simple maneuver like a turn becomes difficult. There is not enough fin authority left to maintain depth and turn at the same time. Figure 5.27 shows percent fin authority versus speed for the 21UUV at a 2 percent weight-buoyancy mismatch.

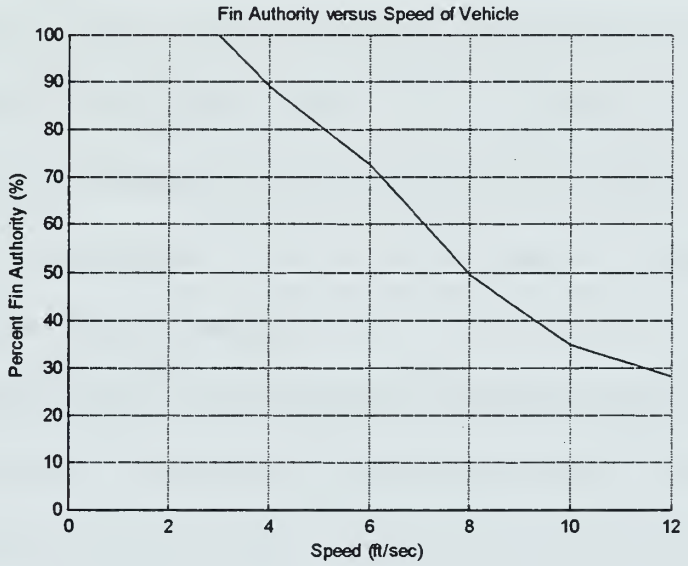

Figure 5.27 Percent Fin Authority versus Speed of Vehicle Expended for 2% mismatch of Weight and Buoyancy

A complete list of simulation runs of the 21UUV model is included in Appendix C.

VI. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

The overall problem of fault detection for 21UUV is complex. Thus, the detection of faults was limited in this study to the primary subsystems of the vehicle, including the diving, steering, roll control and speed control systems. Many current fault detection schemes use static signals. This is done by using "limits and trends" analysis. For dynamic signals, such as a stuck or loose fin, the transient nature of the signal makes limits and trends analysis invalid. Dynamic fault responses can be obtained from servo-error and model based observer residuals.

Model based observer residuals detectors for the diving, steering, and roll control systems were designed and implemented in the *Simulink* model of the 21UUV. In the course of conducting simulations of the 21UUV model, numerous vehicle behaviors were validated. In addition, the model based observer residual detectors have been found to be useful for the detection of fin faults on the vehicle even though they also respond to disturbances and unmodelled dynamic inputs from maneuvering.

The "X" fin configuration of 21UUV provides a level of redundancy, which the control systems (diving, steering, roll, and speed) can use by automatic reconfiguration. Fin 1 is redundant with fin 3 and fin 2 is redundant with fin 4. By implementing normal autopilot control systems, a fault to fin 1 is compensated to some degree by fin 3 without compromising the diving and steering capability of the vehicle. However, a more serious condition arises particularly at low speeds when multiple "faults" are present. Faults considered were weight-buoyancy mismatch and multiple fin faults.

Due to the inherent difficulty of running the 21UUV with perfectly neutral balance, weight and buoyancy mismatch was studied. By setting appropriate threshold levels, the diving observer residuals for heave velocity relative to the water, w_r, and the pitch rate, q, can detect a weight-buoyancy mismatch on the vehicle.

Model based observers may be designed to distinguish fin faults from wave disturbances and fin faults from maneuvering responses. It appears that fin faults can be detected most reliably using the roll observer residuals. Likelihood tests on the residual responses provide further ease of setting detection thresholds. By setting appropriate threshold levels, the roll rate, p, can quickly and robustly determine a fin fault without corruption from wave disturbances or maneuvering responses. The steering observer residuals can not as effectively distinguish a fault from a turn. This is due to the fact that fin faults are at times indistinguishable from the problems of fin saturation which is in the steering control system. Thus, steering observer residuals can best be used to detect faults during nonmaneuvering conditions. Although fin faults are additive in the roll control system, the roll command is always set back to zero, eliminating the direct excitation of residuals by maneuvers.

Control problems arise when 21UUV is forced to go less than 6 feet per second. With a weight-buoyancy mismatch, the vehicle is unable to maintain depth when steering. This is due to the fact that too much fin authority is being used at these lower speeds.

This study has proved that model based observer residuals may be used for the detection of fin faults and weight-buoyancy mismatches regardless of wave disturbances and maneuvering responses, but ultimately, there is a mismatch coupled with more than a

single fin fault that will render the vehicle uncontrollable, aggravated with any weight-buoyancy mismatch.

B. RECOMMENDATIONS

The following recommendations are made in the continuation of this study:

- Use the observer residual for roll rate, p, with maximum likelihood filter as the primary input to fuzzy inference system for detection of fin faults.

- Use a maximum likelihood filter on the model based observer residual signal to get a more distinct difference between maneuvers and fin faults.

- Study the possibility of using diving, steering, and roll observer residuals to detect subsystem (diving, steering, and roll control) faults. This should be done in conjunction with other fault detectors (servo-error, fins-deflection, and wave motion) and tied together via fuzzy inference system.

APPENDIX A. *MATLAB* FILES FOR MODEL BASED OBSERVER DESIGN

This appendix contains *Matlab* files for the design of the diving, steering, and roll control model based observers, respectively.

dive_obs_des.m

```
%  This program was used to design the 21UUV diving observer


%  This program is computed in feet/sec/lbf units
%  the depth controller for a submersible vehicle in forward
%  motion is designed and computed based on SMC methods.
%  Nondimensionalized version of Hydrocoefficients are used
%  and reconstituted into dimensional form. here rho=1.94;L=20;
%  Time is in seconds. V is in feet / sec.
clear
V=6;
t0=0;
tf=300;
rho=1.94;L=20.5;
%
%  Vehicle nondimensional parameters for the DSRV
%  diving response
%
%  this sets up the diving smc design
%
%  open loop system
%
m=88.9518;%slugs
Iy=2632.47;% dimensional already
Iyy=Iy;
Mq=-1.477e-03*0.5*rho*V*L^4;
Mqdot=-7.504e-04*0.5*rho*L^5;
Mw=6.746e-03*0.5*rho*V*L^3;
Mwdot=-1.753e-04*0.5*rho*L^4;
Md=-2.176e-03*0.5*rho*V^2*L^3;
Mth=0.02*m*32.2;
Zq=-2.655e-03*0.5*rho*V*L^3;
Zqdot=-1.753e-04*0.5*rho*L^4;
Zw=-7.406e-03*0.5*rho*V*L^2;
Zwdot=-1.041e-02*0.5*rho*L^3;
Zd=-4.216e-03*0.5*rho*V^2*L^2;


% (ND time is L/V = 20/4 = 5 seconds)

MM=[(m-Zwdot),-Zqdot,0,0;-Mwdot,(Iyy-Mqdot),0,0;0,0,1,0;0,0,0,1];
AA = [Zw,(Zq+m*V),0,0;Mw,Mq,Mth,0;0,1,0,0;1,0,-V,0];
BB = [Zd;Md;0;0];
A=inv(MM)*AA;B=inv(MM)*BB;C=eye(4,4);D=zeros(4,1);
[num,den]=ss2tf(A,B,C,D);p=roots(den);

%  desired closed loop poles for sliding are [-0.4,-0.41,-0.42,0];

% k=place(A,B,[-0.84,-0.8641,-0.7,0]);
k=place(A,B,[-0.4,-0.41,-0.42,0]);
% k=place(A,B,[-0.2,-0.21,-0.22,0]);
```

```
%  closed loop dynamics matrix
Ac=A-B*k;
[m,n]=eig(Ac');

s=m(:,4);

%  Critical Speed for Input reversals
%
% Uc=[-B(1)*(A(2,3)*V^2)/(A(2,1)*B(1)-B(2)*A(1,1))]^0.5
%
%  simulation of closed loop response
%

C=eye(4,4);

%  Observer Gain Matrix

%K=lqe(A,eye(4,4),C,1,eye(4,4).*100);
K=place(A',C',[-0.2,-0.21,-0.22,-0.24]);
%K=place(A',C',[-0.4,-0.41,-0.42,-0.44]);
%K=place(A',C',[-0.6,-0.61,-0.62,-0.64]);
eig(A-K'*C)

%  Dive Observer Design

Ao=A-K'*C;
Bo=[B,K'];
Co=-C;
Do=[zeros(4,1),eye(4,4)]

end
```

steer_obs_des.m

```
%  This program was used to design the 21UUV steering observer

%  This program is computed in feet/sec/lbf units
%  the steering controller for a submersible vehicle in forward
%  motion is designed and computed based on SMC methods.
%  Nondimensionalized version of Hydrocoefficients are used
%  and reconstituted into dimensional form. here rho=1.94;L=20;
%  Time is in seconds. V is in feet / sec.
clear
V=6;
t0=0;
tf=300;
rho=1.94;L=20.5;
%
%  Vehicle nondimensional parameters for the DSRV
%  steering response
%
%  this sets up the steering smc design
%
%  open loop system
%
m=88.9518;%slugs
Iy=2632.47;% dimensional alread
Iz=Iy;
Xud=-1.667e-04; Yvd=-1.041e-02; Nvd=1.753e-04;
Zwd=-1.041e-02; Mwd=-1.753e-04; Zqd=-1.753e-04;  Mqd=-7.504e-04;
Yrd=1.753e-04;  Nrd=-7.504e-04; Xuu=-8.348e-04;
Yv=-7.406e-03;  Nv=-6.746e-03;  Zw=-7.406e-03;   Mw=6.746e-03;

%increase linear roll damping *10
Kp=-2.423e-06;
Zq=-2.655e-03;  Mq=-1.477e-03;   Yr=2.655e-03;   Nr=-1.477e-03;
Xvv=4.073e-03;  Zvp=-1.041e-02; Mvp=-1.753e-04;
Xvr=1.012e-02;  Xww=4.073e-03;
Ywp=1.041e-02; Nwp=-1.753e-04; Xwq=-1.012e-02;
Ypq=1.753e-04;  Npq=-7.503e-04; Zpr=1.753e-04;   Mpr=7.503e-04;
Xqq=4.982e-05;  Xrr=4.982e-05;  ZDS=-4.216e-03;  MDS=-2.176e-03;
XWDS=2.226e-03;   XQDS=1.148e-03;  XDSDS=-1.429e-03;
YDR=4.216e-03;    NDR=-2.176e-03;   XVDR=-2.226e-03;  XRDR=1.148e-03;
XDRDR=-1.429e-03; KDA=9.674e-05;   XDADA=-2.858e-03;
KPHI2=0;
KPHI4=3.506E-05;

%long. center of rotation off the body
Zqaq=-4.121E-03;  Mqaq=-1.770E-03;
Zwaq=-2.270E-02;  Mwaq=-9.487E-03;
Zw2=-5.917E-02;      Mw2=-2.474E-02;
Zw3=3.750E-03;    Mw3=1.132E-02;
Zw4=1.293E-01;    Mw4=5.463E-02;
Zwaw=-5.643E-02;  Mwaw=-1.135E-02;
Zq2=-4.743E-03;      Mq2=-1.607E-03;
Zqaw=-3.712E-02;  Mqaw=-1.107E-02;
```

```
%lat center of rotation on the body
Yvav=-5.643E-02;  Nvav=1.135E-02;
Yvar=2.270E-02;   Nvar=-9.487E-03;
Yr2=-4.743E-03;   Nr2=1.607E-03;

%lat center of rotation off the body
Yrar=4.121E-03;   Nrar=-1.770E-03;
Yrav=-3.712E-02;  Nrav=1.107E-02;
Yv2=-5.917E-02;   Nv2=2.474E-02;
Yv3=-3.750e-3;   Nv3=1.132e-2;
Yv4=1.293e-1;    Nv4=-5.463e-2;

%Dimensional design for steering autopilot
Nvdot=Nvd*0.5*rho*L^4;
Nrdot=Nrd*0.5*rho*L^5;
Yvdot=Yvd*0.5*rho*L^3;
Yrdot=Yrd*0.5*rho*L^4;
Nv=Nv*0.5*rho*V*L^3;
Nr=Nr*0.5*rho*V*L^4;
Yv=Yv*0.5*rho*V*L^2;
Yr=Yr*0.5*rho*V*L^3;
Nd=NDR*0.5*rho*V^2*L^3;
Yd=YDR*0.5*rho*V^2*L^2;

MM=[(m-Yvdot) -Yrdot 0;-Nvdot (Iz-Nrdot) 0;0 0 1];
AA=[Yv (Yr-m*V) 0;Nv Nr 0; 0 1 0];
BB=[Yd;Nd;0];
A=inv(MM)*AA;B=inv(MM)*BB;C=[0,0,1];D=0;
[num,den]=ss2tf(A,B,C,D);z=roots(num);p=roots(den);

%  desired closed loop poles for sliding are [-0.4,-0.41,-0.42,0];

k=place(A,B,[-0.4,-0.41,0]);
%  closed loop dynamics matrix
Ac=A-B*k;
[m,n]=eig(Ac');
s=m(:,3);

%  Observer Gain Matrix

C=eye(3,3);
%K=lqe(A,eye(3,3),C,1,eye(4,4).*100);

K=place(A',C',[-0.2,-0.21,-0.22]);
%K=place(A',C',[-0.4,-0.41,-0.42]);
%K=place(A',C',[-0.6,-0.61,-0.62]);
%K=place(A',C',[-0.8,-0.81,-0.82])
eig(A-K'*C)

% Steering Observer Design

Ao=A-K'*C;
Bo=[B,K'];
Co=-C;
Do=[zeros(3,1),eye(3,3)]

end
```

```
%  This program was used to design the 21UUV steering observer

%  This program is computed in feet/sec/lbf units
%  the roll controller for a submersible vehicle in forward
%  motion is designed and computed based on SMC methods.
%  Nondimensionalized version of Hydrocoefficients are used
%  and reconstituted into dimensional form. here rho=1.94;L=20;
%  Time is in seconds. V is in feet / sec.
clear
V=6;
t0=0;
tf=300;
rho=1.94;L=20.5;
%
%  Vehicle nondimensional parameters for the DSRV
%  roll response
%
%  this sets up the roll smc design
%
%  open loop system
%
m=88.9518;%slugs
Iy=2632.47;% dimensional already
Iz=Iy;
Xud=-1.667e-04; Yvd=-1.041e-02; Nvd=1.753e-04;
Zwd=-1.041e-02; Mwd=-1.753e-04; Zqd=-1.753e-04;  Mqd=-7.504e-04;
Yrd=1.753e-04;  Nrd=-7.504e-04; Xuu=-8.348e-04;
Yv=-7.406e-03;  Nv=-6.746e-03;  Zw=-7.406e-03;   Mw=6.746e-03;

%increase linear roll damping *10
Kp=-2.423e-06;
Zq=-2.655e-03;  Mq=-1.477e-03;   Yr=2.655e-03;   Nr=-1.477e-03;
Xvv=4.073e-03;  Zvp=-1.041e-02; Mvp=-1.753e-04;
Xvr=1.012e-02;  Xww=4.073e-03;
Ywp=1.041e-02; Nwp=-1.753e-04; Xwq=-1.012e-02;
Ypq=1.753e-04;  Npq=-7.503e-04; Zpr=1.753e-04;   Mpr=7.503e-04;
Xqq=4.982e-05;  Xrr=4.982e-05;  ZDS=-4.216e-03;  MDS=-2.176e-03;
XWDS=2.226e-03;   XQDS=1.148e-03;  XDSDS=-1.429e-03;
YDR=4.216e-03;    NDR=-2.176e-03;   XVDR=-2.226e-03;  XRDR=1.148e-03;
XDRDR=-1.429e-03; KDA=9.674e-05;   XDADA=-2.858e-03;KPHI2=0;
KPHI4=3.506E-05;

%long. center of rotation off the body
Zqaq=-4.121E-03;  Mqaq=-1.770E-03;
Zwaq=-2.270E-02;  Mwaq=-9.487E-03;
Zw2=-5.917E-02;      Mw2=-2.474E-02;
Zw3=3.750E-03;    Mw3=1.132E-02;
Zw4=1.293E-01;    Mw4=5.463E-02;
Zwaw=-5.643E-02;  Mwaw=-1.135E-02;
Zq2=-4.743E-03;      Mq2=-1.607E-03;
Zqaw=-3.712E-02;  Mqaw=-1.107E-02;

%lat center of rotation on the body
Yvav=-5.643E-02;  Nvav=1.135E-02;
```

```
Yvar=2.270E-02;   Nvar=-9.487E-03;
Yr2=-4.743E-03;   Nr2=1.607E-03;

%lat center of rotation off the body
Yrar=4.121E-03;   Nrar=-1.770E-03;
Yrav=-3.712E-02;  Nrav=1.107E-02;
Yv2=-5.917E-02;   Nv2=2.474E-02;
Yv3=-3.750e-3;   Nv3=1.132e-2;
Yv4=1.293e-1;    Nv4=-5.463e-2;

%Dimensional design for steering autopilot

Nvdot=Nvd*0.5*rho*L^4;
Nrdot=Nrd*0.5*rho*L^5;
Yvdot=Yvd*0.5*rho*L^3;
Yrdot=Yrd*0.5*rho*L^4;Nv=Nv*0.5*rho*V*L^3;
Nr=Nr*0.5*rho*V*L^4;
Yv=Yv*0.5*rho*V*L^2;
Yr=Yr*0.5*rho*V*L^3;
Nd=NDR*0.5*rho*V^2*L^3;
Yd=YDR*0.5*rho*V^2*L^2;
MM=[(m-Yvdot) -Yrdot 0;-Nvdot (Iz-Nrdot) 0;0 0 1];
AA=[Yv (Yr-m*V) 0;Nv Nr 0; 0 1 0];
BB=[Yd;Nd;0];
A=inv(MM)*AA;B=inv(MM)*BB;C=[0,0,1];D=0;
[num,den]=ss2tf(A,B,C,D);z=roots(num);p=roots(den);

%  desired closed loop poles for sliding are [-0.4,-0.41,-0.42,0];

k=place(A,B,[-0.4,-0.41,0]);
%  closed loop dynamics matrix
Ac=A-B*k;
[m,n]=eig(Ac');
s=m(:,3);

%Roll Controller design
Ix=32.7826;
Kd=KDA*0.5*rho*V^2*L^3;
Ar=[0,1;-0.02*88.95*32.2/Ix,0];
Br=[0;Kd/Ix];
kr=place(Ar,Br,[-1.4,0]);
Acr=Ar-Br*kr;
[m,n]=eig(Acr');
s=m(:,2);

%  Observer Gain Matrix
Cr=eye(2,2);
Kc=place(Ar',Cr',[-1.40,-1.41]);
%Kc=place(Ar',Cr',[-0.60,-0.61]);

%  Roll Observer Design
Aor=Ar-Kc'*Cr;
Bor=[Br,Kc'];
Cor=-Cr
Dor=[zeros(2,1),eye(2,2)]

end
```

APPENDIX B. *MATLAB* FILES FOR 21UUV COMPUTER MODEL

This appendix contains *Matlab* files used in the 21UUV computer model. All files were originally written by Healey and Miguel and modified to meet the needs of specific simulations.

```
function
[du,dv,dw,dp,dq,dr]=body_vel(u,v,w,p,q,r,phi,theta,DS,DR,DA,Fxp,Mxp,
SFxh,SFyh,SFzh,SMxh,SMyh,SMzh,Mxsin,Fzcf,Mycf,Fycf,Mzcf)


global rho m l Wg B Zg Ix Iy Iz xt Diam;

global Xud Yvd Nvd Zwd  Mwd Zqd Mqd Yrd Nrd Xuu Yv Nv Zw  Mw Kp Zq Mq
Yr Nr;
global Xvv Zvp Mvp Xvr Xww Ywp Nwp Xwq Ypq Npq Zpr  Mpr Xqq  Xrr;
global ZDS  MDS XWDS XQDS  XDSDS YDR NDR XVDR XRDR XDRDR KDA  XDADA;


      t1 = rho*rho;
      t2 = SMyh*t1;
      t3 = l*l;
      t4 = t3*t3;
      t5 = t4*t4;
      t6 = t5*Mqd;
      t7 = t2*t6;
      t8 = SFzh*Zwd;
      t9 = sin(theta);
      t10 = t9*B;
      t13 = Wg*t9;
      t14 = t13*t2;
      t15 = t5*Mwd;
      t16 = SFzh*Zqd;
      t19 = Zg*Zg;
      t20 = t19*Wg;
      t23 = SFzh*rho;
      t24 = t3*l;
      t25 = t24*Zwd;
      t26 = t23*t25;
      t31 = SMyh*rho;
      t32 = t4*l;
      t33 = t32*Mqd;
      t34 = t31*t33;
      t36 = m*Iy;
      t39 = t31*t4*Mwd;
      t40 = m*Zg;
      t41 = cos(theta);
      t42 = cos(phi);
      t43 = t41*t42;
      t44 = t43*B;
      t47 = m*m;
      t50 = t31*t32;
      t52 = B*m;
      t55 = t1*rho;
      t56 = SMyh*t55;
      t57 = t5*t3;
      t58 = t57*Mwd;
      t60 = t56*t58*SFzh;
      t61 = Zqd*SFxh;
```

```
      t63 = u*w;
      t64 = t63*DS;
      t67 = u*u;
      t68 = XDADA*t67;
      t69 = DA*DA;
      t70 = t68*t69;
      t74 = v*u;
      t75 = t74*DR;
      t78 = XDRDR*t67;
      t79 = DR*DR;
      t80 = t78*t79;
      t83 = t5*t24;
      t84 = t83*Mwd;
      t86 = t56*t84*SFzh;
      t88 = u*r;
      t89 = t88*DR;
      t93 = u*q;
      t94 = t93*DS;
      t97 = t56*t58;
      t98 = SFxh*Xuu;
      t99 = t98*t67;
      t102 = -2.0*t7*t8*t10-2.0*t14*t15*t16+4.0*t20*t9*m*t26-
4.0*t13*Iy*t26-4.0...
*t13*m*t34+8.0*t13*t36-4.0*t39*t40*t44-
8.0*t20*t9*t47+4.0*t50*Mqd*t9*t52+t60*...
t61*XWDS*t64+t60*t61*t70+t60*t61*XVDR*t75+t60*t61*t80+t86*t61*XRDR*t89+
t86*t61*...
XQDS*t94+t97*t16*t99;
      t103 = Xvr*v;
      t104 = t103*r;
      t107 = Xwq*w;
      t108 = t107*q;
      t111 = t5*t4;
      t113 = t56*t111*Mwd;
      t114 = SFxh*Xrr;
      t115 = r*r;
      t116 = t114*t115;
      t119 = t31*t4;
      t120 = Mwd*t47;
      t121 = p*p;
      t125 = q*q;
      t129 = t6*t8;
      t133 = Iy*SFzh;
      t134 = t133*rho;
      t137 = t2*t15;
      t140 = t43*Wg;
      t143 = t5*l;
      t145 = t2*t143*Mqd;
      t146 = t115*m;
      t147 = t114*t146;
      t149 = SFxh*Xvr;
      t150 = v*r;
      t154 = t4*t3;
      t157 = t2*t154*Mwd*m;
      t158 = Zg*SFzh;
      t160 = Zw*u*w;
      t163 = SFzh*Fzcf;
```

```
      t166 = t4*t24;
      t167 = t166*Mwd;
      t169 = t2*t167*m;
      t171 = Zvp*v*p;
      t175 = Zq*u*q;
      t178 =
t86*t61*t104+t86*t61*t108+t113*t16*t116+4.0*t119*t120*t19*t121+4.0...
*t119*t120*t19*t125+2.0*t14*t129-
8.0*Iy*t9*t52+4.0*t134*t25*t10+2.0*t137*t16*...
t10+4.0*t39*t40*t140+2.0*t145*t147+2.0*t7*t149*t150*m+2.0*t157*t158*t16
0+4.0*...
t39*t40*t163+2.0*t169*t158*t171+2.0*t169*t158*t175;
      t181 = ZDS*t67*DS;
      t184 = t47*Zg;
      t185 = v*p;
      t186 = t184*t185;
      t188 = t184*t93;
      t190 = SFxh*Xqq;
      t191 = t190*t125;
      t196 = Zqd*m;
      t198 = Zg*p*r;
      t201 = XDSDS*t67;
      t202 = DS*DS;
      t203 = t201*t202;
      t206 = SFxh*Xww;
      t207 = w*w;
      t208 = t206*t207;
      t211 = SFxh*Fxp;
      t214 = SFxh*Xvv;
      t215 = v*v;
      t216 = t214*t215;
      t219 = t166*Mqd;
      t220 = t2*t219;
      t221 = SFxh*XDADA;
      t222 = t67*t69;
      t227 = t2*t219*SFxh;
      t228 = XWDS*u;
      t231 = t228*w*DS*m;
      t233 = XVDR*v;
      t236 = t233*u*DR*m;
      t238 = SFxh*XDRDR;
      t239 = t67*t79;
      t244 = t2*t6*SFxh;
      t245 = XRDR*u;
      t248 = t245*r*DR*m;
      t250 = XQDS*u;
      t253 = t250*q*DS*m;
      t255 = t67*m;
      t256 = t98*t255;
      t258 = 2.0*t157*t158*t181-
4.0*t39*t186+4.0*t39*t188+t113*t16*t191-2.0*t2*...
t15*SFzh*t196*t198+t60*t61*t203+t97*t16*t208+2.0*t137*t16*t211+t97*t16*
t216+2.0...
*t220*t221*t222*m+2.0*t227*t231+2.0*t227*t236+2.0*t220*t238*t239*m+2.0*
t244*...
t248+2.0*t244*t253+2.0*t220*t256;
      t259 = SFxh*Xwq;
```

```
      t260 = w*q;
      t266 = t57*Mqd;
      t268 = t56*t266*SFzh;
      t269 = Zwd*SFxh;
      t273 = t269*t70;
      t278 = t269*t80;
      t281 = Fxp*m;
      t286 = t56*t83*Mqd*SFzh;
      t290 = t56*t266;
      t293 = t269*t108;
      t295 = t269*t104;
      t298 = t56*t111*Mqd;
      t301 = t125*m;
      t302 = t190*t301;
      t304 = p*r;
      t307 = SFxh*XDSDS;
      t308 = t67*t202;
      t312 = t207*m;
      t313 = t206*t312;
      t315 = t215*m;
      t316 = t214*t315;
      t318 = 2.0*t7*t259*t260*m-2.0*t7*t8*t211-t268*t269*XWDS*t64-
t268*t273-...
t268*t269*XVDR*t75-t268*t278+4.0*t50*Mqd*SFxh*t281-t286*t269*XQDS*t94-
t290*t8*...
t99-t286*t293-t286*t295-t298*t8*t116+2.0*t145*t302-
4.0*t34*t184*t304+2.0*t220*...
t307*t308*m+2.0*t220*t313+2.0*t220*t316;
      t321 = Iy*rho;
      t323 = t321*t3*SFxh;
      t324 = t202*m;
      t327 = t321*t3;
      t333 = t69*m;
      t337 = t79*m;
      t340 = t24*SFxh;
      t341 = t321*t340;
      t348 = q*m;
      t351 = r*m;
      t354 = t321*t4;
      t358 = -4.0*t323*t201*t324-4.0*t327*t313-4.0*t327*t316-
4.0*t323*t231-8.0*...
Iy*SFxh*t281-4.0*t323*t68*t333-4.0*t323*t236-4.0*t323*t78*t337-
4.0*t341*t248...
-4.0*t341*t253-4.0*t327*t256-t286*t269*XRDR*t89-4.0*t341*t107*t348-
4.0*t341*...
t103*t351-4.0*t354*t147-t298*t8*t191;
      t359 = t269*t203;
      t363 = Zwd*m;
      t364 = t363*t198;
      t370 = t1*t32;
      t371 = t133*t370;
      t372 = Xvv*t215;
      t378 = t133*t370*Zwd;
      t379 = SFxh*XWDS;
      t383 = SFxh*XVDR;
      t387 = t1*t154;
      t389 = t133*t387*Zwd;
```

```
      t390 = SFxh*XRDR;
      t393 = SFxh*XQDS;
      t396 = Xuu*t67;
      t399 = t133*t387;
      t402 = t1*t166;
      t403 = t133*t402;
      t404 = Xrr*t115;
      t408 = -t268*t359+2.0*t2*t6*SFzh*t364-t290*t8*t208-
t290*t8*t216+2.0*t371*...
t269*t372+4.0*t134*t25*t211+2.0*t378*t379*t64+2.0*t371*t273+2.0*t378*t3
83*t75+...
2.0*t371*t278+2.0*t389*t390*t89+2.0*t389*t393*t94+2.0*t371*t269*t396+2.
0*t399*...
t293+2.0*t399*t295+2.0*t403*t269*t404-4.0*t354*t302;
      t410 = t47*Iy;
      t412 = m*w;
      t413 = q*SMyh;
      t414 = t412*t413;
      t415 = t1*t5;
      t416 = Mwd*SFzh;
      t418 = t415*t416*Zqd;
      t420 = m*v;
      t422 = t420*r*SMyh;
      t423 = Mqd*SFzh;
      t425 = t415*t423*Zwd;
      t427 = t47*v;
      t430 = r*Iy;
      t434 = t47*t19;
      t437 = t47*m;
      t438 = t437*t19;
      t441 = Xqq*t125;
      t444 = rho*t24;
      t448 = Xww*t207;
      t451 = SMyh*Mycf;
      t457 = DS*m;
      t458 = t158*Zwd;
      t462 = 8.0*t410*t198-2.0*t414*t418-
2.0*t422*t425+4.0*t427*r*t34+4.0*t420*...
t430*t26-8.0*t427*t430-
4.0*t434*t150*t26+8.0*t438*t150+2.0*t422*t418+2.0*t403*...
t269*t441-4.0*t133*t444*t364+2.0*t371*t359+2.0*t371*t269*t448-
4.0*t451*t40*t26...
-2.0*t2*t154*MDS*t67*t457*t458+8.0*t451*t184;
      t465 = t67*DS;
      t477 = Iz*m;
      t478 = t304*t477;
      t479 = t444*Zwd;
      t480 = t158*t479;
      t482 = Iz*t47;
      t488 = Ix*m;
      t489 = t304*t488;
      t492 = q*Iy;
      t495 = t47*w;
      t519 = 4.0*t31*t24*MDS*t465*t184-
2.0*t2*t166*Mvp*v*p*m*t458+4.0*t31*t4*...
Mvp*t186-4.0*t478*t480+8.0*t304*t482*Zg-
8.0*t304*Ix*t47*Zg+4.0*t489*t480+2.0*...
```

```
t414*t425-4.0*t412*t492*t26-
4.0*t495*q*t34+4.0*t434*t260*t26+8.0*t495*t492-8.0*...
t438*t260-2.0*t2*t154*Mw*u*t412*t458+4.0*t31*t24*Mw*t63*t184-
2.0*t2*t166*Mq*u*...
t348*t458+4.0*t31*t4*Mq*t188;
      t527 = SFxh*Xud;
      t534 = t47*SMyh;
      t535 = rho*t32;
      t538 = m*SMyh;
      t541 = SFxh*rho;
      t542 = t541*t24;
      t543 = Xud*Iy;
      t546 = SFxh*t1;
      t551 = Xud*SMyh;
      t555 = SFxh*t55;
      t556 = t83*Xud;
      t558 = SMyh*Mqd;
      t561 = -2.0*t2*t5*t416*t196+t56*t84*t16*t527-
8.0*t438+4.0*t434*SFzh*t479+...
8.0*t410-4.0*t36*SFzh*t479-4.0*t534*t535*Mqd+2.0*t538*t1*t129-
4.0*t542*t543*m+...
2.0*t546*t154*t543*t8+2.0*t546*t5*t551*Mqd*m-t555*t556*t558*t8;
      t562 = 1/t561;
      du = -(t102+t178+t258+t318+t358+t408+t462+t519)*t562;
      t564 = q*r;
      t565 = Iz*Iz;
      t566 = t564*t565;
      t568 = SMzh*t1;
      t571 = w*p;
      t572 = SFyh*Yrd;
      t573 = t572*Ix;
      t576 = t5*Nrd;
      t577 = t568*t576;
      t579 = t40*SMxh*Mxsin;
      t581 = SMzh*rho;
      t582 = t32*Nrd;
      t583 = t581*t582;
      t584 = t564*Ix;
      t585 = t40*t584;
      t587 = t434*t571;
      t589 = t564*Iz;
      t590 = t40*t589;
      t592 = t564*Iy;
      t593 = t40*t592;
      t598 = Zg*SMxh;
      t601 = Iz*rho;
      t608 = t598*KDA*t67*DA;
      t610 = SMzh*Mzcf;
      t612 = rho*t4;
      t613 = Yrd*Ix;
      t614 = t612*t613;
      t616 = -4.0*t40*t566+t568*t5*Nwp*t571*t573-
t577*t579+2.0*t583*t585+2.0*...
t583*t587+2.0*t583*t590-2.0*t583*t593-
4.0*t482*t19*w*p+4.0*t477*t598*Mxp+2.0*...
t601*t24*t579+2.0*t601*t24*m*t608+2.0*t610*SFyh*t614;
      t617 = t143*Nrd;
```

```
      t622 = t598*Kp*u*p;
      t624 = t434*t88;
      t627 = t88*Ix;
      t628 = SFyh*Yr*t627;
      t632 = t67*DR;
      t636 = Nrd*SFyh;
      t637 = Fycf*Ix;
      t644 = p*Ix;
      t645 = t412*t644;
      t647 = p*q;
      t648 = Ix*Ix;
      t650 = SFyh*rho;
      t652 = t650*t4*Yrd;
      t655 = t568*t166*Nrd;
      t658 = SFyh*YDR*t632*Ix;
      t666 = SFyh*Yv*t74*Ix;
      t671 = -t568*t617*m*t622-2.0*t583*t624-
t577*t628+t568*t166*NDR*t632*t573...
-2.0*t581*t32*t636*t637+t568*t5*Nr*t88*t573-
2.0*t583*t645+2.0*t647*t648*t652-...
t655*t658+t568*t143*Npq*t647*t573-t655*t666+4.0*t482*t19*u*r;
      t673 = t24*SFyh;
      t674 = t601*t673;
      t675 = Yr*u;
      t676 = r*Ix;
      t683 = Iy*SFyh;
      t686 = t3*SFyh;
      t687 = t601*t686;
      t688 = Yv*u;
      t692 = Ywp*w;
      t699 = t477*Zg;
      t701 = Iz*SFyh;
      t703 = t4*SFyh;
      t705 = Ypq*p;
      t709 = t571*Ix;
      t712 = 2.0*t674*t675*t676+t568*t166*Nv*t74*t573-
2.0*t647*t683*t614+2.0*...
t687*t688*v*Ix+2.0*t674*t692*t644-
4.0*t477*t627+2.0*t601*t4*m*t622+4.0*t699*...
t592+4.0*t701*t637+2.0*t601*t703*t705*q*Ix+4.0*t477*t709-4.0*t699*t584;
      t713 = YDR*t67;
      t717 = Iz*t41;
      t718 = sin(phi);
      t725 = t41*t718;
      t727 = t725*Wg*Ix;
      t730 = t725*B*Ix;
      t734 = Wg*t41*t718;
      t735 = m*t19*t734;
      t739 = SMxh*Mxp;
      t740 = t40*t739;
      t746 = SFyh*Ywp*t709;
      t750 = t647*Ix;
      t751 = SFyh*Ypq*t750;
      t753 = m*u;
      t754 = t753*t676;
      t756 = 2.0*t687*t713*DR*Ix-
4.0*t717*t718*B*Ix+4.0*t717*t718*Wg*Ix-2.0*...
```

```
t583*t727+2.0*t583*t730+2.0*t583*t735-4.0*t477*t19*t734-2.0*t583*t740-
t568*t576...
*m*t608-t577*t746-t568*t617*t751+2.0*t583*t754;
      t761 = t535*Nrd;
      t763 = t650*t24;
      t764 = Yvd*Ix;
      t767 = SFyh*t1;
      t768 = t767*t5;
      t769 = SMzh*Nrd;
      t775 = SMzh*Nvd;
      t779 = 1/(-4.0*t488*Iz+2.0*t488*SMzh*t761+2.0*t763*t764*Iz-
t768*t764*t769...
+4.0*t434*Iz-2.0*t434*SMzh*t761+t768*t613*t775);
      dv = -(t616+t671+t712+t756)*t779;
      t781 = SFzh*t55;
      t784 = t781*t57*Zqd*SFxh;
      t786 = MDS*t67*DS;
      t787 = t551*t786;
      t789 = SFzh*t1;
      t790 = t166*Zqd;
      t791 = t789*t790;
      t793 = t527*t304*Iz;
      t797 = t781*t83*Zqd*SFxh;
      t799 = Mvp*v*p;
      t800 = t551*t799;
      t803 = t23*t4*Zqd;
      t806 = Mw*u*w;
      t807 = t551*t806;
      t809 = t527*t451;
      t812 = Mq*u*q;
      t813 = t551*t812;
      t815 = t538*t786;
      t818 = t437*t19*Zg;
      t822 = t789*t5*Zqd;
      t823 = t538*t812;
      t825 = t784*t787+2.0*t791*t793+t797*t800-
4.0*t803*t478+t784*t807+2.0*t791...
*t809+t797*t813-2.0*t791*t815+8.0*t818*t121+8.0*t818*t125-
2.0*t822*t823;
      t827 = t154*Zqd;
      t828 = t789*t827;
      t830 = t98*t255*Zg;
      t833 = t789*t790*SFxh;
      t835 = t457*Zg;
      t836 = t250*q*t835;
      t838 = Xud*m;
      t841 = t838*Zg*v*r;
      t844 = t103*t351*Zg;
      t847 = t114*t146*Zg;
      t850 = t107*t348*Zg;
      t852 = t538*t806;
      t855 = t214*t315*Zg;
      t858 = t206*t312*Zg;
      t861 = t789*t827*SFxh;
      t863 = t201*t324*Zg;
      t865 =
4.0*t803*t489+2.0*t828*t830+2.0*t833*t836+2.0*t833*t841+2.0*t833*...
```

```
t844+2.0*t822*t847+2.0*t833*t850-
2.0*t791*t852+2.0*t828*t855+2.0*t828*t858+2.0*...
t861*t863;
      t868 = t546*t5*Xud;
      t871 = t211*t40;
      t874 = t68*t333*Zg;
      t877 = t228*w*t835;
      t881 = DR*m*Zg;
      t882 = t233*u*t881;
      t885 = t78*t337*Zg;
      t890 = t538*t799;
      t894 = t838*Zg*w*q;
      t897 = t245*r*t881;
      t899 = t434*t304;
      t901 = -
2.0*t868*t558*t140+4.0*t803*t871+2.0*t861*t874+2.0*t861*t877+2.0*...
t861*t882+2.0*t861*t885-4.0*t23*t4*t196*t451-2.0*t822*t890-
2.0*t833*t894+2.0*...
t833*t897-4.0*t803*t899;
      t902 = t434*t23;
      t904 = t24*Zq*t93;
      t907 = t3*Zw*t63;
      t911 = t3*ZDS*t465;
      t915 = t527*t304*Ix;
      t917 = t301*Zg;
      t918 = t190*t917;
      t920 = Zg*t121;
      t924 = Zg*t125;
      t926 = 4.0*t902*t904+4.0*t902*t907-
8.0*t438*t185+4.0*t902*t911+8.0*t438*...
t93-2.0*t791*t915+2.0*t822*t918-8.0*t410*t920-
8.0*t410*t93+8.0*t410*t185-8.0*...
t410*t924;
      t931 = t24*Zvp*t185;
      t933 = t538*t402;
      t934 = t423*t160;
      t936 = t538*t415;
      t937 = t423*t171;
      t943 = t541*t24*Xud;
      t947 = t546*t154*Xud;
      t952 = t423*t181;
      t954 = t534*rho;
      t959 =
8.0*t434*t163+4.0*t902*t931+2.0*t933*t934+2.0*t936*t937+4.0*t538*...
rho*t33*t163+4.0*t943*t36*t920+2.0*t947*t133*t175+2.0*t947*t133*t171+2.
0*t933*...
t952+4.0*t954*t33*t93-4.0*t954*t33*t185;
      t962 = t423*t175;
      t966 = t36*t23;
      t973 = t546*t32*Xud;
      t980 = 4.0*t954*t33*t920+2.0*t936*t962+4.0*t954*t33*t924-
4.0*t966*t907...
-8.0*t36*t163-4.0*t966*t931-4.0*t966*t911-
4.0*t966*t904+2.0*t973*t133*t160+4.0*...
t542*t543*t163+2.0*t973*t133*t181;
      t989 = t555*t556*SMyh;
      t994 = t555*t57*Xud*SMyh;
```

```
      t1007 = t10*t40;
      t1009 = 4.0*t943*t36*t93-4.0*t943*t36*t185+4.0*t943*t36*t924-
t989*t962-...
t989*t937-t994*t952-2.0*t868*t558*t40*t121+2.0*t868*t558*t420*p-
2.0*t868*t558*...
t753*q-2.0*t868*t558*t917+4.0*t803*t1007;
      t1014 = Zg*Wg;
      t1016 = t527*t1014*t9;
      t1021 = Iy*t41;
      t1022 = t42*B;
      t1025 = t538*t535;
      t1026 = Mqd*t41;
      t1029 = t42*Wg;
      t1036 = 8.0*t434*t140-8.0*t434*t44+2.0*t868*t558*t44-
2.0*t791*t1016-t994*...
t934-2.0*t868*t558*t163-4.0*t943*t1021*t1022-
4.0*t1025*t1026*t1022+4.0*t1025*...
t1026*t1029+8.0*t36*t44-8.0*t36*t140+4.0*t943*t1021*t1029;
      dw = -t562*(t825+t865+t901+t926+t959+t980+t1009+t1036);
      t1041 = t24*Yvd;
      t1042 = t650*t1041;
      t1046 = t40*t571*SFyh;
      t1048 = Yvd*SMzh*Nrd;
      t1049 = t415*t1048;
      t1052 = Yrd*SMzh*Nvd;
      t1053 = t415*t1052;
      t1057 = t739*t767;
      t1058 = t5*Yvd;
      t1063 = t650*t1041*Iz;
      t1065 = t564*t701;
      t1067 = t564*t683;
      t1074 = Yvd*Iz;
      t1075 = t444*t1074;
      t1080 = -4.0*t566*t1042-8.0*t739*t477+2.0*t1046*t1049-
2.0*t1046*t1053+4.0...
*t739*m*t583-2.0*t1057*t1058*t769-
4.0*t40*t571*t1063+2.0*t1065*t1049+2.0*t1067*...
t1053-4.0*t564*t477*t583-
2.0*t1067*t1049+8.0*t564*t565*m+4.0*t1067*t1075+4.0*...
t564*t36*t583-2.0*t1065*t1053;
      t1081 = SMxh*t55;
      t1084 = t1081*t83*KDA*t67;
      t1085 = DA*SFyh;
      t1091 = SMxh*t1;
      t1093 = Mxsin*m;
      t1102 = t40*t601;
      t1103 = t675*r;
      t1106 = t40*t647;
      t1107 = t612*Yrd;
      t1110 = t713*DR;
      t1113 = SMxh*rho;
      t1117 = t688*v;
      t1123 = t40*t568*t166;
      t1124 = Nv*u;
      t1133 = t725*B;
      t1137 =
t1084*t1085*t1052+2.0*t1057*t5*Yrd*t775+2.0*t1091*t5*t1093*t769+...
```

```
2.0*t1091*t154*Mxsin*SFyh*t1074+4.0*t739*SFyh*t1075-
4.0*t1102*t673*t1103+4.0*...
t1106*t683*t1107-4.0*t1102*t686*t1110-4.0*t1113*t24*t1093*Iz-
4.0*t1102*t686*...
t1117-4.0*t40*t610*t652-
2.0*t1123*t1124*v*SFyh*Yrd+2.0*t1123*t636*t1117+2.0*...
t1123*t636*t1110+8.0*t699*t1133-4.0*t1014*t725*t1063;
      t1141 = t67*DA;
      t1142 = SFyh*Yvd;
      t1143 = t1142*Iz;
      t1151 = m*SMzh*Nrd;
      t1157 = t1081*t83*Mxsin;
      t1161 = t40*t88*SFyh;
      t1169 = u*p;
      t1182 = t692*p;
      t1186 = t705*q;
      t1192 = 2.0*t1091*t154*KDA*t1141*t1143-
t1084*t1085*t1048+2.0*t1091*t5*KDA...
*t1141*t1151+4.0*t40*t88*t1063-t1157*t1142*t769-2.0*t1161*t1049-
4.0*t1113*t24*...
KDA*t1141*t477+2.0*t1091*t166*Kp*t1169*t1143+t1157*t572*t775-
4.0*t1113*t4*Kp*...
t1169*t477+2.0*t1091*t143*Kp*t1169*t1151-
4.0*t1102*t673*t1182+8.0*t434*t589-4.0...
*t1102*t703*t1186-8.0*t40*t701*Fycf+2.0*t1161*t1053;
      t1195 = t1081*t111*Kp*u;
      t1196 = p*SFyh;
      t1205 = t40*t568*t143;
      t1209 = t40*t568*t5;
      t1214 = NDR*t67;
      t1219 = Nr*u;
      t1224 = t40*t581;
      t1230 = t434*t581;
      t1233 = Npq*p;
      t1238 = Nwp*w;
      t1243 = t1014*t725*SFyh;
      t1248 = -t1195*t1196*t1048+t1195*t1196*t1052-
4.0*t1106*Ix*SFyh*t1107+2.0*...
t1205*t636*t1186+2.0*t1209*t636*t1182-8.0*t564*t36*Iz-
2.0*t1123*t1214*DR*SFyh*...
Yrd-
2.0*t1209*t1219*r*SFyh*Yrd+4.0*t1224*t582*SFyh*Fycf+2.0*t1209*t636*t110
3...
-4.0*t1230*t582*t564-2.0*t1205*t1233*q*SFyh*Yrd-
2.0*t1209*t1238*t1196*Yrd-2.0*...
t1243*t1053-4.0*t1224*t582*t1133+2.0*t1243*t1049;
      dp = (t1080+t1137+t1192+t1248)*t779/2;
      t1257 = Mwd*SFxh*Xud;
      t1263 = SMyh*Mwd*m;
      t1273 = p*SMyh;
      t1297 = t402*t1257;
      t1301 = t612*Mwd*m;
      t1306 = -
t781*t143*Zw*u*w*SMyh*t1257+2.0*t789*t154*Zw*t63*t1263+2.0*t789*...
t166*Zvp*t185*t1263-t781*t57*Zvp*v*t1273*t1257-
t781*t57*Zq*u*t413*t1257+2.0*...
```

```
t789*t166*Zq*t93*t1263-
t781*t143*ZDS*t67*DS*SMyh*t1257+2.0*t789*t154*ZDS*t465*...
t1263-2.0*t40*t121*SMyh*t1297+4.0*t163*SMyh*t1301-
2.0*t163*t2*t167*t527;
      t1308 = t612*SFxh;
      t1311 = t154*Zwd;
      t1312 = t789*t1311;
      t1316 = t781*t57*Zwd*SFxh;
      t1318 = rho*t3;
      t1323 = t1318*SFxh;
      t1330 = DS*t47*Zg;
      t1339 = 8.0*t534*Mycf-
4.0*t1308*t441*t184+2.0*t1312*t793+t1316*t800-4.0*...
t1318*t307*t308*t184+8.0*t438*t304-4.0*t1323*t372*t184-
4.0*t1323*t448*t184-4.0*...
t1318*t379*t63*t1330-4.0*t1318*t221*t222*t184-4.0*t1318*t238*t239*t184;
      t1341 = t184*t260;
      t1345 = DR*t47*Zg;
      t1348 = t184*t150;
      t1355 = t612*t47;
      t1366 = t444*t47;
      t1372 = 4.0*t943*t1341-4.0*t444*t390*t88*t1345-4.0*t943*t1348-
4.0*t1323*...
t396*t184-4.0*t444*t393*t93*t1330+4.0*t1355*SMyh*Mvp*t185-
4.0*t444*t149*t1348...
-4.0*t444*t259*t1341+4.0*t1355*SMyh*Mq*t93+4.0*t1366*SMyh*Mw*t63-
4.0*t1308*t404...
*t184;
      t1376 = t47*p;
      t1377 = r*Iz;
      t1381 = t43*Wg*SMyh;
      t1384 = m*rho;
      t1385 = t1384*t340;
      t1390 = t43*B*SMyh;
      t1399 = 4.0*t1366*SMyh*MDS*t465+8.0*t1376*t1377-
8.0*t1376*t676+4.0*t1381*...
t1301-8.0*t10*t184+4.0*t1385*Xud*Zg*t13-
4.0*t1390*t1301+4.0*t26*t1007+2.0*t1390...
*t1297-2.0*t1312*t1016-4.0*t1318*t383*t74*t1345-8.0*t211*t184;
      t1404 = t781*t143*Zwd*SFxh;
      t1409 = m*t1;
      t1411 = t1409*t166*SFxh;
      t1418 = t1409*t154*SFxh;
      t1420 = Xud*p;
      t1428 = t1404*t787+2.0*t1312*t809-2.0*t1381*t1297+t1316*t813-
2.0*t1411*...
t813+2.0*t420*t1273*t1297+4.0*t184*t121*t39-
2.0*t1418*t807+4.0*t1385*t1420*t676...
+4.0*t184*t125*t39-2.0*t40*t125*SMyh*t1297;
      t1436 = t32*Zwd;
      t1437 = t789*t1436;
      t1441 = t789*t1436*SFxh;
      t1445 = t789*t166*Zwd;
      t1451 = 4.0*t47*u*q*t39-2.0*t753*t413*t1297-
4.0*t427*p*t39+2.0*t1437*t855...
+2.0*t1437*t858+2.0*t1441*t863-4.0*t26*t899+2.0*t1445*t918-
2.0*t1312*t915-4.0*...
```

```
t1385*t1420*t1377-2.0*t1411*t800;
      t1457 = t789*t1311*SFxh;
      t1467 = -2.0*t1418*t787-
4.0*t1384*t24*t809+2.0*t1457*t897+2.0*t1441*t885+...
2.0*t1441*t882+2.0*t1441*t874+2.0*t1445*t847+2.0*t1457*t844+2.0*t1457*t
850+2.0*...
t1437*t830+2.0*t1457*t836;
      t1482 = 2.0*t1457*t841-2.0*t1457*t894-2.0*t1445*t890-
4.0*t23*t24*t363*...
t451-2.0*t1445*t823-2.0*t1312*t852-
2.0*t1312*t815+t1404*t807+4.0*t26*t489-4.0*...
t26*t478+2.0*t1441*t877+4.0*t26*t871;
      dq = (t1306+t1339+t1372+t1399+t1428+t1451+t1467+t1482)*t562;
      t1486 = t166*Nvd;
      t1491 = t581*t4*Nvd;
      t1494 = t568*t1486;
      t1497 = t568*t154*Nvd;
      t1503 = t767*t154*Yvd;
      t1504 = Ix*SMzh;
      t1513 = t767*t166*Yvd;
      t1517 = -t568*t1486*m*t608-2.0*t1491*t593+2.0*t1491*t590-
t1494*t628-t1497...
*t666-t1494*t746-
t1497*t658+2.0*t1491*t735+t1503*t1504*t1124*v+t767*t1058*t1504...
*t1233*q+t1513*t1504*t1219*r;
      t1521 = t5*Nvd;
      t1532 = t647*Iy;
      t1538 = t4*Nwp*t571;
      t1540 = t1503*t1504*t1214*DR-
t568*t1521*m*t622+t1513*t1504*t1238*p-t568*...
t1521*t751+2.0*t1491*t585-4.0*t488*t610+4.0*t488*t1532-2.0*t1491*t740-
t1494*...
t579+2.0*t1491*t587+2.0*t1230*t1538;
      t1543 = t4*Nr*t88;
      t1547 = t24*NDR*t632;
      t1550 = t32*Npq*t647;
      t1559 = t24*Nv*t74;
      t1561 = t488*t581;
      t1566 =
2.0*t1230*t1543+4.0*t434*t610+2.0*t1230*t1547+2.0*t1230*t1550+2.0...
*t763*t764*t610+4.0*t434*t750-4.0*t434*t1532-
2.0*t1042*t644*t492+2.0*t1230*...
t1559-2.0*t1561*t1538+2.0*t763*Yvd*t648*t647;
      t1582 = -2.0*t1561*t1547-2.0*t1561*t1550-2.0*t1491*t727-
4.0*m*t648*t647...
-2.0*t1561*t1559-2.0*t1561*t1543+2.0*t1491*t730-
2.0*t1491*t645+2.0*t1491*t754...
-2.0*t581*t4*Nvd*SFyh*t637-2.0*t1491*t624;
      dr = t779*(t1517+t1540+t1566+t1582);
```

control_design.m

```
%  This program is computed in feet/sec/lbf units
%  the depth controller for a submersible vehicle in forward
%  motion is designed and computed based on SMC methods.
%  Nondimensionalized version of Hydrocoefficients are used
%  and reconstituted into dimensional form. here rho=1.94;L=20;
%  Time is in seconds. V is in feet / sec.
clear
V=6;
t0=0;
tf=300;
rho=1.94;L=20.5;
%
%  Vehicle nondimensional parameters for the DSRV
%  diving response
%
%  this sets up the diving smc design
%
%  open loop system
%  DIVE CONTROL
m=88.9518;%slugs
Iyy=2632.47;% dimensional already

Mq=-1.477e-03*0.5*rho*V*L^4;
Mqdot=-7.504e-04*0.5*rho*L^5;
Mw=6.746e-03*0.5*rho*V*L^3;
Mwdot=-1.753e-04*0.5*rho*L^4;
Md=-2.176e-03*0.5*rho*V^2*L^3;
Mth=0.02*m*32.2;
Zq=-2.655e-03*0.5*rho*V*L^3;
Zqdot=-1.753e-04*0.5*rho*L^4;
Zw=-7.406e-03*0.5*rho*V*L^2;
Zwdot=-1.041e-02*0.5*rho*L^3;
Zd=-4.216e-03*0.5*rho*V^2*L^2;


% (ND time is L/V = 20/4 = 5 seconds)

MM=[(m-Zwdot),-Zqdot,0,0;-Mwdot,(Iyy-Mqdot),0,0;0,0,1,0;0,0,0,1];
AA = [Zw,(Zq+m*V),0,0;Mw,Mq,Mth,0;0,1,0,0;1,0,-V,0];
BB = [Zd;Md;0;0];
A=inv(MM)*AA;B=inv(MM)*BB;C=[0,0,0,1];D=0;
[num,den]=ss2tf(A,B,C,D);z=roots(num);p=roots(den);

%  desired closed loop poles for sliding are [-0.4,-0.41,-0.42,0];

k=place(A,B,[-0.4,-0.41,-0.42,0]);
%  closed loop dynamics matrix
Ac=A-B*k;
[m,n]=eig(Ac');

s=m(:,4);

%  Critical Speed for Input reversals
%
```

```
% Uc=[-B(1)*(A(2,3)*V^2)/(A(2,1)*B(1)-B(2)*A(1,1))]^0.5
%
%  simulation of closed loop response
%
s


%Steering Control design

Iz=Iy;

%
Xud=-1.667e-04; Yvd=-1.041e-02; Nvd=1.753e-04;
Zwd=-1.041e-02; Mwd=-1.753e-04; Zqd=-1.753e-04;  Mqd=-7.504e-04;
Yrd=1.753e-04;  Nrd=-7.504e-04; Xuu=-8.348e-04;
Yv=-7.406e-03;  Nv=-6.746e-03;  Zw=-7.406e-03;   Mw=6.746e-03;

%increase linear roll damping *10
Kp=-2.423e-06;
Zq=-2.655e-03;  Mq=-1.477e-03;   Yr=2.655e-03;   Nr=-1.477e-03;
Xvv=4.073e-03;  Zvp=-1.041e-02; Mvp=-1.753e-04;


Xvr=1.012e-02;  Xww=4.073e-03;
Ywp=1.041e-02; Nwp=-1.753e-04; Xwq=-1.012e-02;
Ypq=1.753e-04;  Npq=-7.503e-04; Zpr=1.753e-04;   Mpr=7.503e-04;
Xqq=4.982e-05;  Xrr=4.982e-05;  ZDS=;  MDS=-2.176e-03;
XWDS=2.226e-03;   XQDS=1.148e-03;  XDSDS=-1.429e-03;
YDR=4.216e-03;    NDR=-2.176e-03;   XVDR=-2.226e-03;  XRDR=1.148e-03;
XDRDR=-1.429e-03; KDA=9.674e-05;   XDADA=-2.858e-03;

KPHI2=0;
KPHI4=3.506E-05;

%long. center of rotation off the body
Zqaq=-4.121E-03;  Mqaq=-1.770E-03;
Zwaq=-2.270E-02;  Mwaq=-9.487E-03;
Zw2=-5.917E-02;      Mw2=-2.474E-02;
Zw3=3.750E-03;    Mw3=1.132E-02;
Zw4=1.293E-01;    Mw4=5.463E-02;

Zwaw=-5.643E-02;  Mwaw=-1.135E-02;
Zq2=-4.743E-03;      Mq2=-1.607E-03;
Zqaw=-3.712E-02;  Mqaw=-1.107E-02;

%lat center of rotation on the body
Yvav=-5.643E-02;  Nvav=1.135E-02;
Yvar=2.270E-02;   Nvar=-9.487E-03;
Yr2=-4.743E-03;   Nr2=1.607E-03;

%lat center of rotation off the body
Yrar=4.121E-03;   Nrar=-1.770E-03;
Yrav=-3.712E-02;  Nrav=1.107E-02;
Yv2=-5.917E-02;   Nv2=2.474E-02;
Yv3=-3.750e-3;   Nv3=1.132e-2;
Yv4=1.293e-1;    Nv4=-5.463e-2;
%    Dimensional design for steering autopilot
```

```
Nvdot=Nvd*0.5*rho*L^4;
Nrdot=Nrd*0.5*rho*L^5;
Yvdot=Yvd*0.5*rho*L^3;
Yrdot=Yrd*0.5*rho*L^4;

Nv=Nv*0.5*rho*V*L^3;
Nr=Nr*0.5*rho*V*L^4;
Yv=Yv*0.5*rho*V*L^2;
Yr=Yr*0.5*rho*V*L^3;
Nd=NDR*0.5*rho*V^2*L^3;
Yd=YDR*0.5*rho*V^2*L^2;


MM=[(m-Yvdot) -Yrdot 0;-Nvdot (Iz-Nrdot) 0;0 0 1];
AA=[Yv (Yr-m*V) 0;Nv Nr 0; 0 1 0];
BB=[Yd;Nd;0];
A=inv(MM)*AA;B=inv(MM)*BB;C=[0,0,1];D=0;
[num,den]=ss2tf(A,B,C,D);z=roots(num);p=roots(den);

%  desired closed loop poles for sliding are [-0.4,-0.41,-0.42,0];

k=place(A,B,[-0.4,-0.41,0]);
%  closed loop dynamics matrix
Ac=A-B*k;
[m,n]=eig(Ac');
s=m(:,3);

% Roll Controller DESIGN
%  Ix
Ix=32.7826;Kpdot=0.0;Kp=-2.423e-06*0.5*rho*V*L^4;
Kphi=-m*32.2*0.02;
%zg=0.02'
Ka=0.5*V^2*rho*L^3*9.674e-05;
A=[0,1;Kp/Ix, Kphi/Ix];B=[0;Ka/Ix];
pdes=[-1.5,0];
k=place(A,B,pdes); % k=[-0.0835   -0.3002]
Ac=A-B*k;
[m,n]=eig(Ac');
s=m(:,2); $ s'=[0,1]
```

euler2body6d.m

```
function output=euler2body6d(Dx,Dy,Dz,Dphi,Dtheta,Dpsi,phi,theta,psi)

% euler to body velocities transformation - generated automaticly with
maple - inertial.ms

      t1 = cos(psi);
      t2 = cos(theta);
      t5 = sin(psi);
      t8 = sin(theta);
      Ubody = t1*t2*Dx+t5*t2*Dy-t8*Dz;
      t10 = Dx*t1;
      t11 = sin(phi);
      t12 = t8*t11;
      t14 = Dx*t5;
      t15 = cos(phi);
      t17 = Dy*t5;
      t19 = Dy*t1;
      Vbody = t10*t12-t14*t15+t17*t12+t19*t15+t11*t2*Dz;
      t23 = t8*t15;
      Wbody = t10*t23+t14*t11+t17*t23-t19*t11+t15*t2*Dz;


      Pbody = Dphi-sin(theta)*Dpsi;
      t3 = cos(phi);
      t5 = sin(phi);
      t6 = cos(theta);
      Qbody = t3*Dtheta+t5*t6*Dpsi;
      Rbody = -t5*Dtheta+t3*t6*Dpsi;

output=[Ubody,Vbody,Wbody,Pbody,Qbody,Rbody];
```

faultsgen.m

```
function [sys, x0] = faultsgen(t,X,in,flag,mode0,FaultM,Faultfile)

global faults findex;

if abs(flag) == 2,    % return of discrete state
   event_hit=faults(findex  ,1)==t;
   if event_hit,
      sys=X;
      sys(faults(findex,2) )= faults(findex  ,3)
      faults
      findex=findex+1
   else
      sys=X;
   end

elseif abs(flag) == 3 % Return systems output
            % (vector of signals that control actuators function mode)
   sys=X;

elseif abs(flag) == 4 % return the next time in which will occur some
fault
   sys=faults(findex  ,1);
elseif flag  == 0,
   x0=mode0;
   numOutputs = length(mode0);   % dynamic number of outputs
   numStates=numOutputs ;

   if nargin<7,
      FaultM
      [faults,I]=sort(FaultM);
      faults(:,2)=FaultM(I(:,2),2);
      faults(:,3)=FaultM(I(:,3),3);
      faults(length(faults),:)=[10000,10,10];%insert add line to matrix
                     % so that findex not exceed matrix size
   elseif nargin==7,
      F = csvread(Faultfile);
      [faults,I]=sort(F);
      faults(:,2)=F(I(:,1),2);
      faults(:,3)=F(I(:,1),3);
      faults(length(faults),:)=[10000,10,10];   % insert an add line to
matrix
                     % so that findex not exceed matrix size
   else
      error('wrong number of parameter to faultsgen  s-function')
   end

   findex=1;

   sys=[0,numStates,numOutputs,0,0,0]; % 0 continuous states, n
discrete, n outputs, 0 inputs
else
   sys=[];
   f=flag;
end
```

inertial.m

```
function [inert_out, x0] =
inertial(t,X,in,flag,xini,yini,zini,phiini,thetaini,psiini)

%global T lin_vel_body DR cpsi ctheta cphi ttheta p q r ;



if abs(flag) == 1,              % Return state rates

%  T=zeros(3,3); iner=zeros(1,6);  lin_vel_body=zeros(3,1);
%  p=zeros(1,1); q=zeros(1,1); r=zeros(1,1);
%  phi=zeros(1,1); theta=zeros(1,1); psi=zeros(1,1);
%  cpsi=zeros(1,1);  spsi=zeros(1,1); ctheta=zeros(1,1);
stheta=zeros(1,1); cphi=zeros(1,1);  sphi=zeros(1,1);
%  DR=zeros(3,1);
%  dphi=zeros(1,1); dtheta=zeros(1,1); dpsi=zeros(1,1);
%  lin_vel_body=in(1:3);
%  ang_vel_body=in(4:6);

   u=in(1);v=in(2);w=in(3);
   p=in(4);q=in(5);r=in(6);
   phi=X(4);
   theta=X(5);
   psi=X(6);

   [dx,dy,dz,dphi, dtheta,dpsi]=inertial_eq(u,v,w,p,q,r,phi,theta,psi);

  inert_out=[dx,dy,dz,dphi, dtheta,dpsi] ;%          +WAVE*Swave;

 % inert_out2=euler2body6([u,v,w,p,q,r]+WAVE*Swave);
 % inert_out=[inert_out1,inert_out2];

elseif abs(flag) == 3,
   inert_out =X;

elseif flag == 0,
        %Return initial conditions
        inert_out = [6,0,6,6,0,0];    % 6 continuous states (inertial
state,
            % 0 discrete,
            % 6 outputs (state),
            % 6 inputs (body rates)
        x0 = [xini,yini,zini,phiini,thetaini,psiini];
%  x0 = [0,0,0,0.1,0,0];

else
        inert_out = [];
        f=flag;
end
```

inertial_eq.m

```
function
[dx,dy,dz,dphi,dtheta,dpsi]=inertial_eq(u,v,w,p,q,r,phi,theta,psi)

% euler angles transformation - generated automaticly with maple -
inertial.ms
      cpsi = cos(psi);
      spsi = sin(psi);
      ctheta = cos(theta);
      stheta = sin(theta);
      cphi = cos(phi);
      sphi = sin(phi);
      t3 = v*cpsi;
      t4 = stheta*sphi;
      t6 = v*spsi;
      t8 = w*cpsi;
      t9 = stheta*cphi;
      t11 = w*spsi;
      dx = cpsi*ctheta*u+t3*t4-t6*cphi+t8*t9+t11*sphi;
      dy = spsi*ctheta*u+t6*t4+t3*cphi+t11*t9-t8*sphi;
      dz = -stheta*u+ctheta*sphi*v+ctheta*cphi*w;
      t28 = 1/ctheta;
      dphi = (p*ctheta+t4*q+t9*r)*t28;
      dtheta = cphi*q-sphi*r;
      dpsi = (sphi*q+cphi*r)*t28;
```

maxllike.m

```
% this program was originally written by R. Christi, NPS

%likelihood ratio test
%t=[0:0.1:100];
%f=rand(1,1001);
%m=mean(f);s=std(f);f=f-m;


%for i=1:length(t),
%   if i>400,f(i)=f(i)+1*s;end; %3 sigma sudden jump in residual
%  end;
%
%figure(1),clf,plot(f)

%   Sjk=d^2/c
[phi,g]=c2d(Aor,Bor(:,1),0.1);x=zeros(2,(length(t)+1));
for i=1:length(t); x(:,i+1)=phi*x(:,i)+g;end;
 rho=x(2,:);
%r=zeros(1, length(t));rho=r;rho(1)=0;
% generate residual signal excited by wn only
%for i=2:length(t),
   %r(i)=1.8006*r(i-1)-0.8187*r(i-2)+0.0905*f(i)-0.0905*f(i-1);
   %   rho(i)=1.8006*rho(i-1)-0.8187*rho(i-2);
%  r(i)=0.9*r(i-1)+0.1*f(i-1);%filtered signal noise +fault
 %  rho(i)=0.9*rho(i-1)+0.1*1;%filtered signal no noise + anticipated
unit fault
%end;

%figure(1),clf
%figure(1),clf,plot(t,r)

[phi,g]=c2d(Aor,Bor(:,1),0.1);x=zeros(2,(length(t)+1));
for i=1:length(t); x(:,i+1)=phi*x(:,i)+g;end;
 rho=x(2,:);
M=50;N=M-1;
d=zeros(length(t),M);l=d;c=zeros(1,M);
for k=M:length(t),
      for j=1:M;
         c(j)=rho((k-j+1):k)*rho((k-j+1):k)'; %AutoCorrel unit fault
         d(k,j)=rho((k-j+1):k)*roll_obs_res((k-j+1):k,2); %cross cor
resid with unit
         l(k,j)=0.5*d(k,j)^2/(c(j)); % increase shows fault level
      end;
      L(k)=2*max(l(k,:));fhat(k)=d(k,M)/c(M);
      g(k)=max(roll_obs_res(k-N:k).^2-(roll_obs_res(k-N:k)-fhat(k-
N:k)).^2);
end;

figure(2),clf,
plot(t,r,'r',t,fhat,'m',t,L,'b'),grid

figure(3),clf,
plot(t,g),grid
```

model.m

```
function [sys, x0] =
model(t,x,IN,flag,uini,vini,wini,pini,qini,rini,amp)

global rho m l Wg B Zg Ix Iy Iz xt Diam;
global log_cross;



if abs(flag) == 1 ,              % Returns state derivative

   inert_state_len=6;
   INERCIAL_STATE=IN(1:inert_state_len);
   FINS_CMDS=IN(inert_state_len +1: inert_state_len +4);
   Prop_n=IN(inert_state_len +5);   %propeller revs (rps)

   u=x(1);      v=x(2);      w=x(3);
   p=x(4);      q=x(5);      r=x(6);

   phi=INERCIAL_STATE(4);
   theta=INERCIAL_STATE(5);
   pz=IN(3);
   delta_s=FINS_CMDS(1);
   delta_r=FINS_CMDS(2);
   delta_a=FINS_CMDS(3);


   Prop_wf=0.8;
   Prop_alfa=0.3;  % ????? A sorte!!!
   Prop_D=2*8.05/12;
   Fxp=rho*Prop_n*abs(Prop_n)*Prop_D^4*0.4-
Prop_alfa*Prop_n*Prop_D^3*rho*u;

  %Fxp=16;
   Mxp=0;

   SMxsin=1;   Sdraglong=1;   Sdraglat=1;
   %SFxh=0;SFyh=0;SFzh=0;SMxh=0;SMyh=0;SMzh=0;
%  SFxh=0;SFyh=0;SFzh=0;SMxh=1;SMyh=1;SMzh=0;
   SFxh=1.0;SFyh=1.0;SFzh=1.0;SMxh=1.0;SMyh=1.0;SMzh=1.0;

   % crossflow evaluation -

[Mxsin,longit,Fzcf,Mycf,lat,Fycf,Mzcf]=crossflow(u,v,w,p,q,r,Sdraglong,
Sdraglat,SMxsin,l,Diam,1);

   log_cross(length(log_cross)+1,:)=[t,Mxsin,longit,Fzcf,Mycf,lat,Fycf,
Mzcf];

% 6DEF EOM - generated automaticly with maple - model.ms
%
   [du,dv,dw,dp,dq,dr]=body_vel(u,v,w,p,q,r,phi,theta,delta_s,delta_r,d
elta_a,Fxp,Mxp,SFxh,SFyh,SFzh,SMxh,SMyh,SMzh, Mxsin, Fzcf, Mycf, Fycf,
Mzcf );
```

```
   [du,dv,dw,dp,dq,dr]=
bodym(u,v,w,p,q,r,phi,theta,delta_s,delta_r,delta_a,Fxp,Mxp,SFxh,SFyh,S
Fzh,SMxh,SMyh,SMzh, Mxsin, Fzcf, Mycf, Fycf, Mzcf );

   sys=[du,dv,dw,dp,dq,dr];




elseif abs(flag) == 3,

   %WAVE_CX=1; WAVE_CY=0.0;
% water current in ft/sec, H=water depth
   H=30;
   WAVE_CX=0.0; WAVE_CY=0;
   WAVE_AMP=amp; WAVE_T=5;
   WAVE_K=(2*pi)^2/(WAVE_T^2*32);            %(2*pi)^2/(WAVE_T^2*32);
   WAVE_V=sqrt(32.2*H);       %= w/k = 2*pi/WAVE_T/k = WAVE_T*32/(2*pi)
;

   WAVE=wavevel1(t,IN(1),IN(3),WAVE_V,WAVE_AMP,WAVE_T,WAVE_CX,WAVE_CY,H
);
%        WAVE(7) = pressure
%Swave=0;
   Xwave= WAVE(1:6)';

   sys
=(x+euler2body6d(Xwave(1),Xwave(2),Xwave(3),Xwave(4),Xwave(5),Xwave(6),
IN(4),IN(5),IN(6))');

        % sys=[sys;WAVE(7)];
   sys=[sys;x(1:6);WAVE(7)];

elseif flag == 0,
        % sys = [6,0,7,11,0,0];    % 6 continuous states , 0 discrete,
7 outputs, 11 inputs
   sys = [6,0,13,11,0,0];    % 6 continuous states , 0 discrete, 13
outputs, 11 inputs
   log_cross=[];

        %Return initial conditions
        x0 = [uini,vini,wini,pini,qini,rini];
%  x0 = [6,0,0,0,0,0];

else
        sys = [];
        f=flag;
end
```

predict.m

```
function [rho] = predict(delta)
%  Matlab script to function as a Eighth-Order Digital Filter for
%       Predicting Future Seaway Elevation Response
%       Tracks and matches Pierson Moskowitz/Pressure Profile
%       Spectrum and predicts responses one full wave length
%
%       Variables
%       h = Fignificant  wave height
%       w = Frequency
%          S = Defines Pierson Moskowitz expression for fully developed
seas
%       t = Time vector
%       zeta = Damping ratio
%       T = Period interval
%       A,B = Continuous plant model
%       Phi,Gamma = Discrete plant model
%       K = Filter Gains
%       error=
%       x2 = Estimate of state vector
%       Y = System's output
%            rho = Cross-correlation coefficient
%            delta is the number of time steps, (i.e. Phi is delta
minus one)

h=3;
w=[0.3:0.05:3];dw=0.05;
[l,ms]=size(w);
S=w;
for i=1:ms
    S(i)=8.1e-3*32.2^2/(w(i)^5)*exp(-33.56/h^2/(w(i)^4));
 end;
ws=[0.3:0.05:3];

lambda=[171.61,146.52,127.62,112.86,100.98,91.21,83.01,76.02,70,64.73,.
..
60.08,55.94,52.23,48.88,45.83,43.05,40.49,38.14,35.96,33.93,32.05,30.29
,...
28.64,27.1,25.66,24.3,23.03,21.84,20.71,19.66,18.67,17.75,16.88,16.07,1
5.30,...
14.59,13.92,13.29,12.7,12.15,11.63,11.15,10.69,10.26,9.86,9.47,9.11,8.7
7,...
8.45,8.15,7.86,7.59,7.33,7.08,6.85];

%  Pressue Measurement Simulation

lambda=lambda.*3.28;
t=[0:0.1:200];
 Y=zeros(1,length(t));
 for i=1:length(ws);
     phi=rand(1,length(ws));phi=phi-mean(phi);
     y(i,:)=(cosh(6*pi/lambda(i))/cosh(40*pi/lambda(i)))...
     *(sqrt(S(i)*2*dw))*cos(ws(i)*t+phi(i)*pi*2);
 end;
```

```
    for j=1:length(ws);
          Y=Y+y(j,:);
    end;

%  Defines Coefficients of transfer function of Eighth-Order Filter

T=0.1;w0=1.28;zeta=0.4;
n=[1/w0,0];d=[(1/w0)^2,2*zeta/w0,1];    % Defines Fourth-Order
Expression
num=conv(n,n);num4=conv(num,num);
den=conv(d,d);den4=conv(den,den);      % Defines transfer function

%  Defines innovater gains for subject filter

[A1,B1,C1,D1]=tf2ss(num4,den4);
[Phi,Gamma]=c2d(A1,B1,T);
K=dlqr(Phi',C1', eye(8,8)*10,.1); eig1=abs(eig(Phi-K'*C1))
x2=zeros(8,length(t));x4=x2;P2=zeros(1,length(t));P6=P2;

%   For loop for closed loop filter

for i=1:length(t)
  x2(:,i+1)=(Phi)*x2(:,i)+K'*(Y(i)-C1*x2(:,i));
  x60(:,i+1)=(Phi^delta)*x2(:,i+1);
  P2(i)=C1*x2(:,i);
  P60(i)=C1*x60(:,i);
end;

for i=1:(length(t)-delta);
 error60(i)=(P60(i)-Y(i+delta));Y60(i)=Y(i+delta);
end;

%  Calculates cross-correlation coefficient between P-M spectrum and
filter

rho=(Y60*P60(1:(2001-delta))'/(std(Y60)*std(P60)))/(2001-delta);
```

start_up.m

```
%function out=start_up()

global log_cross ctr_sig esignals edetection N_com;

global kin_to_unit kft_to_unit kslugs_to_unit klbm_to_unit
klbf_to_unit;
global klbm_d_ft3_to_unit kslug_m_ft2_to_unit ;

global rho m l Wg B Zg Ix Iy Iz xt Diam;
global Xud Yvd Nvd Zwd  Mwd Zqd Mqd Yrd Nrd Xuu Yv Nv Zw  Mw Kp Zq Mq
Yr Nr;
global Xvv Zvp Mvp Xvr Xww Ywp Nwp Xwq Ypq Npq Zpr  Mpr Xqq  Xrr;
global ZDS  MDS XWDS XQDS  XDSDS YDR NDR XVDR XRDR XDRDR KDA  XDADA;
global KPHI2 KPHI4;

%long. center of rotation off/on the body
global Zqaq Mqaq Zwaq Mwaq Zw2 Mw2 Zw3 Mw3 Zw4 Mw4;
global Zwaw Mwaw Zq2 Mq2 Zqaw Mqaw

%lat center of rotation on/off the body
global Yvav Nvav Yvar Nvar Yr2  Nr2;
global Yrar Nrar Yrav Nrav Yv2 Nv2 Yv3 Nv3 Yv4 Nv4;

   % TO use SI
%  kin_to_unit=.0254;      %(to m)
%  kft_to_unit=.3048;      %(to m)
%  kslugs_to_unit=14.5939;    %(to Kg)
%  klbm_to_unit=0.4536;    %(to Kg)
%  klbf_to_unit=4.4482;    %(to N)

   % TO      ft, sec,slugs, lbf
   kin_to_unit=1/12;       %(to ft)
   kft_to_unit=1;       %(to ft)
   kslugs_to_unit=1;    %(to slugs)
   klbm_to_unit=0.4536/14.5939;  %(to slugs)
   klbf_to_unit=1;         %(to lbf)


   %To use other define: kin_to_unit, kft_to_unit, kslugs_to_unit,
   %           klbm_to_unit, klbf_to_unit


   klbm_d_ft3_to_unit = klbm_to_unit/(kft_to_unit^3);
   kslug_m_ft2_to_unit = kslugs_to_unit*kft_to_unit^2;

               % initial values in :
   rho=62.41*klbm_d_ft3_to_unit;    % (lbm/ft^3)

   m=88.9518*kslugs_to_unit;     % (slugs) (1slugs = 14.5939Kg ,
1slug=32.17lbm)
   l=246.0*kin_to_unit;       % (in)
   Wg=2861.9353*klbf_to_unit;    % (lbf) (1N=4.4482 lbf)
   B=2811.9351*klbf_to_unit;     % (lbf)
```

```
%2*
   Zg=2*0.12*kin_to_unit;        % (in)

   Ix=32.7826*kslug_m_ft2_to_unit;   % (slug-ft^2)
   Iy=2632.47*kslug_m_ft2_to_unit;   % (slug-ft^2)
   Iz=2632.47*kslug_m_ft2_to_unit;  % (slug-ft^2)
   xt=115.9979*kin_to_unit-1;
   Diam=20.94*kin_to_unit;
   %parr=[ro,m,l,Wg,B,Zb,Ix,Iy,Iz,xt]


%Nondimensional coeficients
Xud=-1.667e-04; Yvd=-1.041e-02; Nvd=1.753e-04;
Zwd=-1.041e-02; Mwd=-1.753e-04; Zqd=-1.753e-04;  Mqd=-7.504e-04;
Yrd=1.753e-04;  Nrd=-7.504e-04; Xuu=-8.348e-04;
Yv=-7.406e-03;  Nv=-6.746e-03;  Zw=-7.406e-03;   Mw=6.746e-03;

%increase linear roll damping *10
Kp=-2.423e-06;
Zq=-2.655e-03;  Mq=-1.477e-03;   Yr=2.655e-03;   Nr=-1.477e-03;
Xvv=4.073e-03;  Zvp=-1.041e-02; Mvp=-1.753e-04;

Xvr=1.012e-02;  Xww=4.073e-03;
Ywp=1.041e-02; Nwp=-1.753e-04; Xwq=-1.012e-02;
Ypq=1.753e-04;  Npq=-7.503e-04; Zpr=1.753e-04;   Mpr=7.503e-04;
Xqq=4.982e-05;  Xrr=4.982e-05;  ZDS=-4.216e-03;  MDS=-2.176e-03;
XWDS=2.226e-03;   XQDS=1.148e-03;  XDSDS=-1.429e-03;
YDR=4.216e-03;    NDR=-2.176e-03;   XVDR=-2.226e-03;  XRDR=1.148e-03;
XDRDR=-1.429e-03; KDA=9.674e-05;   XDADA=-2.858e-03;

KPHI2=0;
KPHI4=3.506E-05;

%long. center of rotation off the body
Zqaq=-4.121E-03;  Mqaq=-1.770E-03;
Zwaq=-2.270E-02;  Mwaq=-9.487E-03;
Zw2=-5.917E-02;      Mw2=-2.474E-02;
Zw3=3.750E-03;    Mw3=1.132E-02;
Zw4=1.293E-01;    Mw4=5.463E-02;

Zwaw=-5.643E-02;  Mwaw=-1.135E-02;
Zq2=-4.743E-03;      Mq2=-1.607E-03;
Zqaw=-3.712E-02;  Mqaw=-1.107E-02;

%lat center of rotation on the body
Yvav=-5.643E-02;  Nvav=1.135E-02;
Yvar=2.270E-02;   Nvar=-9.487E-03;
Yr2=-4.743E-03;   Nr2=1.607E-03;

%lat center of rotation off the body
Yrar=4.121E-03;   Nrar=-1.770E-03;
Yrav=-3.712E-02;  Nrav=1.107E-02;
Yv2=-5.917E-02;   Nv2=2.474E-02;
Yv3=-3.750e-3;   Nv3=1.132e-2;
Yv4=1.293e-1;    Nv4=-5.463e-2;

%end;
```

wavevel.m

```
function vel= wavevel1(t,x,z,c0,amp,Tw,cx,cy,H)

% Trochoidal wave motion, regular wave train moving in the x-direction
%   v=   wave velocity, (ft/sec)
%   amp=wave amplitude, (ft) = wave heigth /2
%   n=   wave number, (rad/ft) = w^2/g = 4*Pi^2/Tw^2*g (for gravity
wave)
%    - Tw= wave period; n = 2*pi/L; with
L=T*sqrt((g*L/2/pi)*tanh(2*pi*H/L)/2/pi
%  For shallow waters, H<<L:  L=T*sqrt(g*H);
%   U=vehicle speed
%   c0 = wave speed
%   HEad seas only at this time
%   cx= water current flowing in the x-direction (ft/sec)
%   cy= water current flowing in the y-direction (ft/sec)
%   x=  x-global position of the vehicle
%   z=  z-global position of the vehicle
%   t=  time (sec)

rho=1.94;U0=6;
g=32.2; L=Tw*sqrt(g*H);
s1=sinh(2*pi*(H-z)/L);s2=sinh(2*pi*(H/L));
c1=cosh(2*pi*(H-z)/L);c2=cosh(2*pi*(H/L));


dx=2*pi*amp/Tw*c1/s2*cos(2*pi*((x-U0*t)/L-t/Tw))+cx+U0;
dy=cy;
dz=2*pi*amp/Tw*s1/s2*sin(2*pi*((x-U0*t)/L-t/Tw));
[dx,dz]
dxang=  0;
dyang=  0;
dzang=  0;

pz=  rho*g*amp*c1/c2*cos(2*pi*(x/L-t(1-U0/c0)/Tw));

vel=[dx,dy,dz,dxang,dyang,dzang];
```

APPENDIX C. LIST OF SIMULATION RUNS FOR THE 21UUV MODEL

This appendix contains a list of all simulation runs for the 21UUV computer model.

File Name	Duration	Speed	Depth	Course	Waves		Command			Fault		
(*.mat)	(sec)	(ft/sec)	(ft)	(deg)	Dir (deg)	Amp (ft)	Time (sec)	Type	To	Time (sec)	Type	Fin
data_001	50	10	10	0	0	0	NA	NA	NA	NA	NA	NA
data_002	50	8	10	0	0	0	NA	NA	NA	NA	NA	NA
data_003	50	6	10	0	0	0	NA	NA	NA	NA	NA	NA
data_004	50	4	10	0	0	0	NA	NA	NA	NA	NA	NA
data_005	50	2	10	0	0	0	NA	NA	NA	NA	NA	NA
data_006	50	10	10	0	0	2	NA	NA	NA	NA	NA	NA
data_007	50	8	10	0	0	2	NA	NA	NA	NA	NA	NA
data_008	50	6	10	0	0	2	NA	NA	NA	NA	NA	NA
data_009	50	4	10	0	0	2	NA	NA	NA	NA	NA	NA
data_010	50	2	10	0	0	2	NA	NA	NA	NA	NA	NA
data_011	80	10	10	0	0	2	20	Steer	45	NA	NA	NA
data_012	80	8	10	0	0	2	20	Steer	45	NA	NA	NA
data_013	80	6	10	0	0	2	20	Steer	45	NA	NA	NA
data_014	80	4	10	0	0	2	20	Steer	45	NA	NA	NA
data_015	80	2	10	0	0	2	20	Steer	45	NA	NA	NA
data_016	80	10	25	0	0	2	20	Dive	0	NA	NA	NA
data_017	80	8	25	0	0	2	20	Dive	0	NA	NA	NA
data_018	80	6	25	0	0	2	20	Dive	0	NA	NA	NA
data_019	80	4	25	0	0	2	20	Dive	0	NA	NA	NA
data_020	80	2	25	0	0	2	20	Dive	0	NA	NA	NA
data_021	80	10	10	0	0	2	30	Steer	45	20	Stuck	1
data_022	80	8	10	0	0	2	30	Steer	45	20	Stuck	1
data_023	80	6	10	0	0	2	30	Steer	45	20	Stuck	1
data_024	80	4	10	0	0	2	30	Steer	45	20	Stuck	1
data_025	80	2	10	0	0	2	30	Steer	45	20	Stuck	1
data_026	80	6	10	0	0	2	30	Steer	45	20	Loose	1
data_027	80	6	10	0	0	2	30	Steer	45	20	Lim	1
data_028	80	6	10	0	0	2	30	Steer	45	20	Loose	2
data_029	80	6	10	0	0	2	30	Steer	45	20	Stuck	2
data_030	80	6	10	0	0	2	30	Steer	45	20	Lim	2
data_031	80	10	25	0	0	2	30	Dive	0	20	Stuck	2
data_032	80	8	25	0	0	2	30	Dive	0	20	Stuck	2
data_033	80	6	25	0	0	2	30	Dive	0	20	Stuck	2

File Name	Duration	Speed	Depth	Course	Waves		Command			Fault		
(*.mat)	(sec)	(ft/sec)	(ft)	(deg)	Dir (deg)	Amp (ft)	Time (sec)	Type	To	Time (sec)	Type	Fin
data_034	80	4	25	0	0	2	30	Dive	0	20	Stuck	2
data_035	80	2	25	0	0	2	30	Dive	0	20	Stuck	2
data_036	80	6	25	0	0	2	30	Dive	0	20	Loose	2
data_037	80	6	25	0	0	2	30	Dive	0	20	Lim	2
data_038	80	6	25	0	0	2	30	Dive	0	20	Stuck	1
data_039	80	6	25	0	0	2	30	Dive	0	20	Loose	1
data_040	80	6	25	0	0	2	30	Dive	0	20	Lim	1
data_041	100	10	50	0	0	2	NA	NA	NA	50	Stuck	1
										80	Stuck	2
data_042	100	8	10	0	0	0	50	Steer	45	20	Stuck	1
data_043	100	8	25	0	0	0	40	Dive	0	20	Stuck	2
data_044	80	4	10	0	0	0	NA	NA	NA	20	Stuck	1
data_045	80	4	10	0	0	0	NA	NA	NA	20	Stuck	1
										40	Stuck	3
data_046	50	12	10	0	0	2	NA	NA	NA	NA	NA	NA
data_047	100	8	25	0	0	0	40	Dive	10	20	Stuck	2
data_048	100	8	10	0	0	0	50	Steer	30	20	Stuck	1
data_049	100	8	10	0	0	0	20	Dive	25	60	Stuck	2
data_050	100	8	10	0	0	0	20	Steer	45	60	Stuck	1
data_051	50	10	10	0	0	0	NA	NA	NA	20	Stuck	1
data_052	50	8	10	0	0	0	NA	NA	NA	20	Stuck	1
data_053	50	6	10	0	0	0	NA	NA	NA	20	Stuck	1
data_054	50	4	10	0	0	0	NA	NA	NA	20	Stuck	1
data_055	50	6	10	0	0	0	NA	NA	NA	20	Loose	1
data_056	50	6	10	0	0	0	NA	NA	NA	20	Stuck	2
data_057	50	6	10	0	0	0	NA	NA	NA	20	Loose	2
data_058	50	10	10	0	0	2	NA	NA	NA	20	Stuck	1
data_059	50	8	10	0	0	2	NA	NA	NA	20	Stuck	1
data_060	50	6	10	0	0	2	NA	NA	NA	20	Stuck	1
data_061	50	4	10	0	0	2	NA	NA	NA	20	Stuck	1
data_062	50	6	10	0	0	2	NA	NA	NA	20	Loose	1
data_063	50	6	10	0	0	2	NA	NA	NA	20	Stuck	2
data_064	50	6	10	0	0	2	NA	NA	NA	20	Loose	2

LIST OF REFERENCES

Beard, R., 1971, "Failure Accommodation in Linear Systems Through Self Reorganization," Report MVT-71-1, Man Vehicle Laboratory, MIT, Cambridge, Massachusetts.

Bell, D. et al., 1992, "Using Causal Reasoning for Automated Failure Modes & Effects Analysis (FMEA)," Proceedings of the Annual Reliability and Maintainability Symposium, Las Vegas, Nevada, pp. 343-353.

Bellingham, J.G., 1997, "Sampling Strategies for Ocean Observation with Mobile Platforms," Marine Society Journal, Vol. 31, No. 3, pp. 34-37.

Cristi, R., Healey, A.J. and Papoulias, F.A., "Adaptive Sliding Mode Control of Autonomous Underwater Vehicles in the Dive Plane," IEEE Journal of Oceanic Engineering, Vol. 15, No. 7, pp. 225-232.

Fossen, T.I., 1994, "Guidance and Control of Ocean Vehicles," John Wiley & Sons Ltd., ISBN-0471-94113-1.

Gertler, J., 1986, "Failure Detection and Isolation in Complex Plants – A Survey," IFAC Symposium on Microcomputer Applications in Process Control, Istanbul, Turkey.

Healey, A.J., 1998, "Analytical Redundancy and Fuzzy Inference in AUV Fault Detection and Compensation," Naval Postgraduate School, Monterey, California.

Healey, A.J., 1997, Course Notes, Dynamics of Marine Vehicles, Naval Postgraduate School, Monterey, California.

Healey, A.J. and Lienard, D., 1993, "Multivariable Sliding Mode Control for Autonomous Diving and Steering of Unmanned Underwater Vehicles," IEEE Journal of Oceanic Engineering, Vol. 18, No. 3, pp. 327-339.

Healey, A.J., 1992, "A Neural Network Approach to Failure Diagnostics for Underwater Vehicles," Proceedings of 1992 IEEE Conference, Washington, D.C., June, pp. 131-134.

Humphreys, D.E., Smith, N.S. and Watkinson, K.W., 1994 "Hydrodynamic Coefficients and Six Degree-of-Freedom Model for the NUWC UUV," Vehicle Control Technologies Inc., Burke, Virginia.

Hurni, M.A., 1997, "Development of an On-Line Failure Mode Detection and Resolution Algorithm for the Phoenix AUV," Master's Thesis, Naval Postgraduate School, Monterey, California.

Isermann, R., 1984, "Process Fault Detection Based on Modeling and Estimation Methods – A Survey," Automatica, Vol. 20, No.4, pp. 387-404.

Jones, R., 1973, "Failure Detection in Linear Systems," Ph.D Dissertation, Department of Aeronautics and Astronautics, MIT, Cambridge, Massachusetts.

Mangoubi, R.S., Appleby, B.D., Vergese, G.C. and Vandervelde, W.E., 1995, "A Robust Failure Detection and Isolation Algorithm," Proceedings of 34th Conference on Decision and Control, New Orleans, Louisiana, December, pp. 2377-2382.

Marco, D.B. and Healey, A.J., 1996, "Local Area Navigation Using Sonar Feature Extraction and Model Based Predictive Control," Autonomous Underwater Vehicles Laboratory, Naval Postgraduate School, Monterey, California.

Napolitano, M.R., 1991, "New Technique for Aircraft Flight Control Reconfiguration," Journal of Guidance and Control, Vol. 14, No.1, pp. 184-190.

Newland, D.E., 1993, "An Introduction to Random Vibrations, Spectral & Wavelet Analysis," Longman Scientific and Technical, ISBN 0-13-280470-0.

Office of Naval Research, 1997, "Intelligent Control: Fault Detection and Compensation," ONR Workshop, Naval Postgraduate School, Monterey, California.

Patton, R.J., 1997, "Fault Tolerant Control: The 1997 Situation," Proceedings of Safeprocess, Hull, United Kingdom, June, pp. 1033-1055.

Patton, R.J. and Chen, J., 1991, "Robust Fault Detection Using Eigenstructure Assignment: A Tutorial Consideration and Some New Results," Proceedings of 30th Conference on Decision and Control, pp. 2242-2247.

Peng, et. al., 1997, "A Complete Procedure for Residual Generation and Evaluation with Application to a Heat Exchanger," IEEE Transactions on Control Systems Technology, Vol. 5, No. 7, pp. 351-370.

Rauch, H.E., 1995, "Autonomous Control Reconfiguration," IEEE Control Systems, Vol. 10, No.12, pp. 37-47.

Speyer, J.L., 1987, "Detection Filter Design; Spectral Theory and Algorithms," IEEE Transactions on Automatic Control, Vol. 32, No. 7, pp.593-603.

Willsky, A.S., 1976, "A survey of design methods for failure detection in dynamic systems," Automatica, Vol. 12, No. 12, pp. 601-611.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center 2
   8725 John J. Kingman Road, Ste 0944
   Ft. Belvoir, Virginia 22060-6218

2. Dudley Knox Library 2
   Naval Postgraduate School
   411 Dyer Road
   Monterey, California 93943-5101

3. Professor Anthony J. Healey, Code ME/He 2
   Naval Postgraduate School
   700 Dyer Road
   Monterey, California 93943-5101

4. Mechanical Engineering Department Chairman, Code ME 1
   Naval Postgraduate School
   700 Dyer Road
   Monterey, California 93943-5101

5. Naval/Mechanical Engineering Curricular Officer, Code 34 1
   Naval Postgraduate School
   700 Dyer Road
   Monterey, California 93943-5101

6. LT James E. Melvin 3
   4208 St Paul Way, #107
   Concord, California 94518

7. Mr. Dan Steiger, Code 32 1
   Office of Naval Research
   800 Quincy Street
   Arlington, Virginia 22217

8. Mike Keegan, Underwater Vehicles 1
   NUWC, Code 221
   Newport, Rhode Island 02841-5047

9. Mr. C. Hillenbrand 1
   NUWC, Code 221
   Newport, Rhode Island 02841-5047

10. Naomi Leonard 1
Assistant Professor
Department of Mechanical and Aerospace Engineering
D-209A Engineering Quadrangle
Princeton University
Princeton, NJ 08544